Flight Control of a Millimeter-Scale Flapping-Wing Robot

Flight Control of a Millimeter-Scale Flapping-Wing Robot

A dissertation presented
by

Pakpong Chirarattananon

to
The School of Engineering and Applied Sciences
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of

Engineering Sciences

Harvard University
Cambridge, Massachusetts
August 2014

Thesis advisor

Robert J. Wood

Author

Pakpong Chirarattananon

Flight Control of a Millimeter-Scale Flapping-Wing Robot

Abstract

Flying insects display remarkable maneuverability. Unlike typical airplanes, these insects are able to execute an evasive action, rapidly change their flight speed and direction, or leisurely land on flowers buffeted by wind, exhibiting aerodynamic feats unmatched by any state-of-the-art aircraft. By subtly tuning their wing motions, they generate and manipulate unsteady aerodynamic phenomenon that is the basis of their extraordinary maneuverability.

Inspired by these tiny animals, scientists and engineers have pushed the boundaries of technology in many aspects, including meso-scale fabrication, electronics, and artificial intelligence, to develop autonomous millimeter-scale flapping-wing robots. In this thesis, we demonstrate, on real insect-scale robots, that using only an approximate model of the aerodynamics and flight dynamics in combination with conventional tools in nonlinear control, the inherently unstable flapping-wing robot can achieve steady hover. We present the development of flight controllers that gradually enhance the flight precision, allowing the robot to realize increasingly aggressive trajectories, including a highly acrobatic maneuver—perching on a vertical surface, as observed in its natural counterparts. We also demonstrate that these experiments lead to higher fidelity of in-flight aerodynamic models, strengthening our understanding

of the dynamics of the robot and real insects.

Contents

Acknowledgments

First and foremost, I would like to express my gratitude to my PhD advisor, Professor Robert Wood. Not only did Rob give me an opportunity to be a part of this remarkable RoboBees project, but he also has been a truely supportive, encouraging, and inspiring mentor, even in his sabbatical year. It has been a privilege to work with Rob and be one of his graduate students.

I must thank the members and alumni of the Harvard Microrobotics Laboratory, both for being wonderful colleagues and friends in the past three years. A special thanks is due to Kevin Ma. Without his exceptional work on the design and fabrication of the insect-scale robots, my contribution on flight control might be pointless. I would also like to thank my collaborators: Dr Sawyer Fuller, Zhi Ern Teoh, and Professor Nestor Perez-Arancibia (USC).

Finally, this dissertation would not have been possible without my parents, who provided me with education, opportunity, and the continued unconditional support.

Chapter 1

Introduction

1.1 Natural and Artificial Flight

In 2005, the Airbus A380, the world's largest passenger airliner (figure 1.1a), first took off from Toulouse, France, 102 years after the aircraft designed and built by the Wright brothers was successfully airborne for the first time in 1903 (figure 1.1b). Equipped with four turbofan engines, the A380 is sized for a maximum take-off weight over 650 tons, almost $2,000$ times that of the original Wright Flyer (338 kg). Over the course of one hundred years, humans have revolutionized aviation. More than a thousand aircraft models have been designed and built.

Looking at nature for comparison, the largest flying animal that ever lived, *Argentavis magnificens* [35], weighed around $70 - 75$ kg—a mere fraction of the Wright Flyer. The giant bird roamed the windy slopes of the Andes six million years ago. With an 8-meter wingspan, they were the size of small airplanes. Yet, on the other end of the spectrum, the tiniest insects such as fruit flies (*Drosophila melanogaster*)

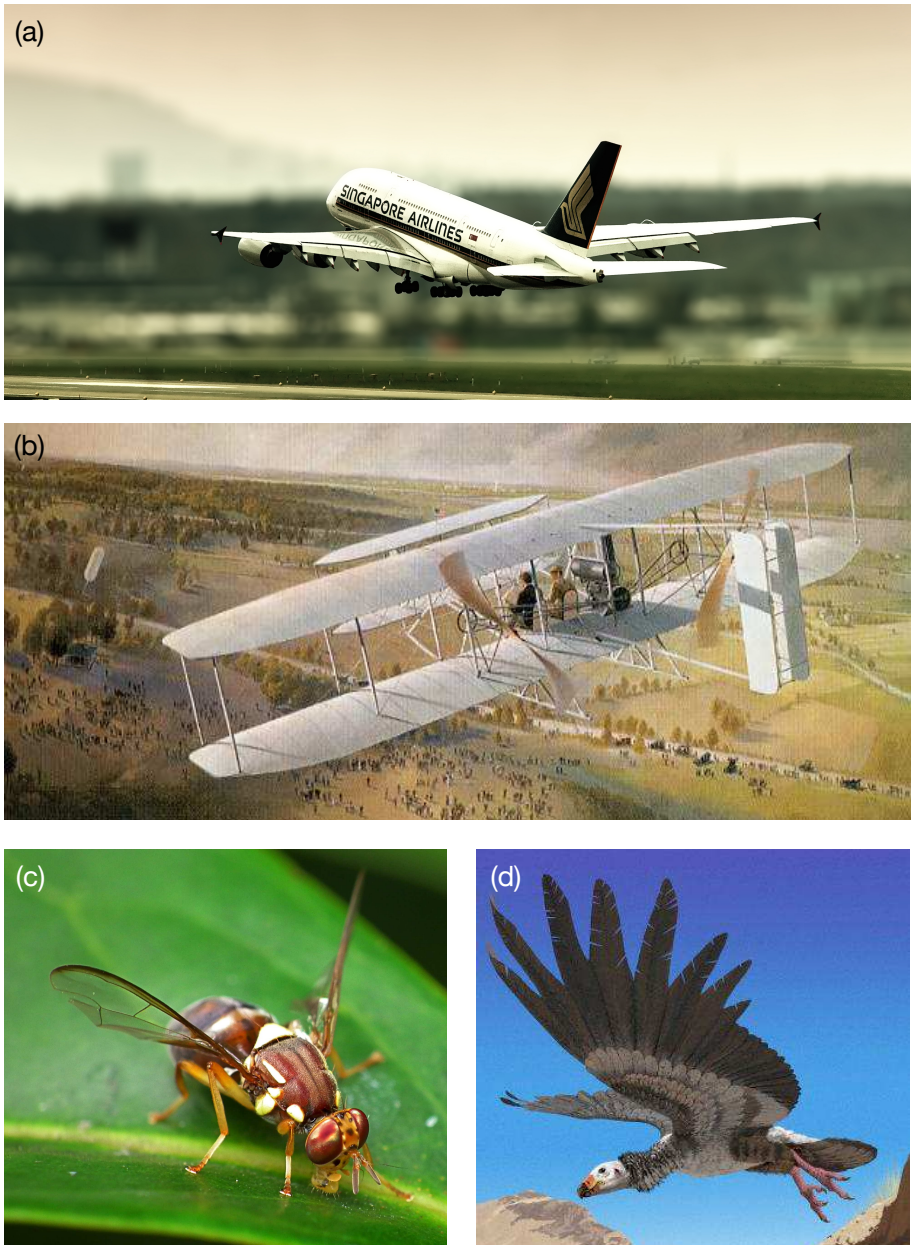

Figure 1.1: (a) Airbus A380 has a wingspan of 80 m with a maximum speed of 945 km per hour. (b) An artistic render of the Wright Flyer, the first successful heavier-than-air powered aircraft. (c) Fruit flies are inherently unstable without feedback control. (d) *Argentavis magnificens* is one of the largest flying birds ever known [35].

weigh around one milligram, and parasitic wasps (*Encarsia formosa*) weigh only a fraction of a milligram [75] on the order of 10^{-12} times lighter than the Airbus A380.

The size difference between artificial fliers and natural fliers is reflected in the lift generation mechanisms for both [79]. Conventional airplanes with fixed wings rely on forward motion relative to the air to generate lift, with thrust produced via propellers or jet engines. At smaller scales and reduced speeds, drags are dominant and fixed wings are no longer an energy efficient method to provide lift for flight. The lift generation of biological fliers—birds and insects—involves various modes of wing motion: sweeping the wings up and down or back and forth, altering the stroke plane, and modulating the angle of attack. In fact, it is believed that *magnificens*, the largest birds, used flapping flight only for short periods and flew mainly by soaring. Flapping becomes more common in smaller birds. Most small birds flap their wings continuously and glide occasionally [78, 68].

1.2 Micro Aerial Vehicles

There are more than $9,000$ species of birds, and smaller fliers, insects, make up approximately 80% of the world's species. These animals maneuver their bodies efficiently through space with exceptional maneuverability unmatched by conventional airplanes. Such superior maneuvering and flight characteristics are primarily the consequences of scaling laws with respect to a vehicle's size [79]. Unsurprisingly, over the last decade, there has been growing interest in *Micro Aerial Vehicles* (MAVs), or flying machines at smaller scales than conventional aircraft.

MAV development must overcome a number of challenges. As the characteristic

length scale shrinks, physics and aerodynamics change. MAVs operate in low Reynolds number regimes, where viscous forces are increasingly important and may dominate inertial forces. At centimeter or smaller scales, airfoil theory, the fundamental design tool of conventional aircraft, is no longer applicable. Researchers have to rely on more fundamental Navier–Stokes equations to explain the aerodynamic phenomena. Unsteady aerodynamics emerge as a solution for lift generation as steady airflow over fixed-wing is no longer sufficient. As a result, flapping-wing flight and rotary-wing flight are promising replacements to fixed-wing systems.

To date, rotary platforms have gained considerable popularity. Multi-rotor designs like quadrotors offers benefits over conventional helicopters that require swashplates and sophisticated control structures. The simple structures of quadrotors renders them easy to model and control. Moreover, they are known to be adept in maneuvering in constrained three-dimensional environments and possess the ability to hover in place. As a consequence, researchers employ quadrotors as platforms for the study of control strategies [59, 60, 63], planning [77], localization [38], and multi-agent systems [86] to name but a few.

1.3 Flight of Flapping-Wing Vehicles

In addition to rotorcraft, flapping-wing flight naturally appears as a biologically inspired solution to small scale artificial flight. Birds and insects utilize unsteady aerodynamics and leading-edge vortices extensively in lift generation. They are model organisms that inspire scientists and engineers to create MAVs that mimic the flapping motion seen in these animals. Some notable examples of artificial flapping-wing de-

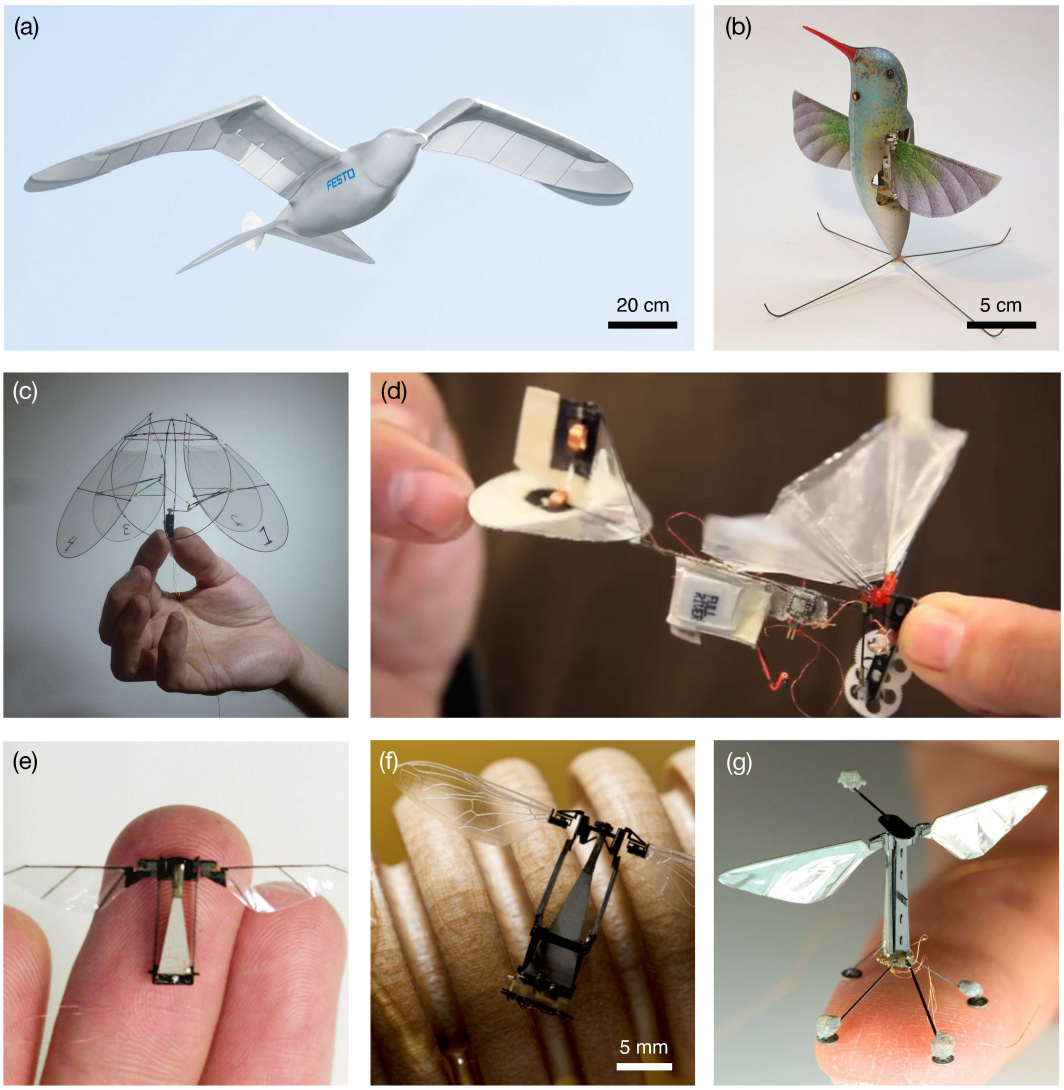

Figure 1.2: (a) Festo Smartbird, a 450-gram autonomous ornithopter created by Festo's Bionic Learning Network [25]. (b) AeroVironment Nano Hummingbird is a remote controlled aircraft that weighs 19 grams [48]. (c) A four-winged robot that flies with a jellyfish-like motion [74]. (d) DelFly Micro has a wingspan of 10 cm [13]. (e) The first insect-scale flying robot able to takeoff [92]. (f) A robotic insect prototype with a bio-inspired torque generation mechanism [29]. (g) A flight-capable flapping-wing vehicle that is employed as a platform for the design of a flight control system in this thesis [58].

vices are shown in figure 1.2. The Festo Smartbird (figure 1.2a) and AeroVironment's Nano Hummingbird (figure 1.2b) are bird-like autonomous vehicles with 2-meter and 16-centimeter wingspans respectively. In both robots, onboard sensors provide feedback to ensure flight stability. On smaller scales, the DelFly Micro in figure 1.2(d) is a passively stable 3-gram ornithopter with a camera and transmitter onboard. Passive stability eliminates the requirement of extensive onboard sensors, simplifying the control problem and enabling the device to be light and small. Passive stability also exists in a centimeter-scale jellyfish-like flying robot in figure 1.2(c) that uses its flapping mechanism to generate propulsion for flight without feedback control. In this case, the 2.1-gram robot is tethered for external power.

1.4 Flapping-Wing Robotic Insects

The agility of flying insects inspires scientist and engineers to create artificial flapping-wing flight. However, these attempts are encountered with myriad engineering challenges, in particular, in meso-scale actuation and manufacturing. Early developments of insect-scale flapping-wing robots were pioneered in the Micromechanical Flying Insect project [95, 3], followed by an early prototype of the Harvard Microrobotic Fly (figure 1.2e)—a 60-mg robotic insect that can produce sufficient thrust for constrained liftoff [92].

The achievement in [92] led to the beginning of the RoboBees project, which seeks to develop a colony of autonomous robotic insects. The effort to create such millimeter-scale robots is motivated by numerous possible applications such as assisted agriculture, reconnaissance, search and rescue in hazardous environments, as

well the potential to aid entomologists in the study of insect flight. This subsequently led to the first flight-capable 83-mg prototype that was able to command body torques using a central power actuator and smaller control actuators resembling insect flight muscles [29]. The robot demonstrated a short successful unconstrained vertical controlled flight in [29].

One of the recent successful prototypes from the RoboBees project was designed, fabricated and first appeared in [58]. The 80-mg MAV shown in figure 1.2(g) is a result of the culmination of research in meso-scale actuation [91] and advances in manufacturing technology [81]. The flapping-wing robot is able to generate body torques and sufficient lift force [58] satisfying the key requirements for flight. Without control, however, the robot cannot achieve flight due to the inherent instability [67, 75].

At insect scales, accomplishing a stable flight is not trivial. Flapping-wing flight entails complex fluid-structure interactions and harnessing unsteady aerodynamics for lift and torque generation. Furthermore, the instability and maneuverability of the robotic insect poses a difficult control problem reminiscent of the control of fighter jets. Nonetheless, among these overwhelming challenges, it promises the potential for high-performance maneuverability nonexistent in conventional aircraft.

1.5 Contributions and Chapter Organization

The goal of the work presented in this dissertation is to design a control system for the millimeter-scale flapping-wing robot in figure 1.2(g). Flapping-wing robots at this scale are, similar to their insect counterparts, inherently unstable without active

feedback control [67]. Here, the control system entails modeling and understanding the dynamics of the insect-scale robot, designing flight controllers to stabilize the vehicle, enabling it to stay in the air as real flying insects. Hovering flight is chosen as the initial target as the dynamics of the robot and its interaction with the surrounding aerodynamics can be substantially simplified. To date, this thesis presents the first integrative design of a flight controller for flying vehicles at this scale that are experimentally validated.

In the next chapter, we review the physical properties of the robot that serves as a primary platform for the flight control experiments in this thesis. The kinematic and dynamic properties of the robot are highlighted. Torque and thrust generation mechanisms are reviewed to construct a plant model for control purposes. Outlines of the translational and rotational dynamics of the robot are also given.

In Chapter 3, we propose a nonlinear flight controller design based on Lyapunov's direct method. The nonlinear properties allow the flight envelope to be expanded—a desired feature for a system lacking thorough characterization with uncertainties. The modular controller emphasizes the attitude dynamics to ensure stability. We demonstrate that the proposed method enables the robot to hover in place within a body length from a setpoint, marking the first unconstrained flight of an insect-scale robot.

Chapter 4 focuses on improving the flight controller based on information obtained from the experiments performed in Chapter 3. We identify uncertain parameters in the system that significantly affect the flight performance. This includes the neutral torque points that determine the trimmed condition of the robot. Adaptive con-

trollers are designed in order to estimate these parameters in an online fashion. The implementation of the adaptive algorithm markedly reduces the position error in hovering flight to approximately one centimeter and evidently decreases the visible body oscillation of the robot in flight.

Towards the goal of performing agile maneuvers as observed in real insects, in Chapter 5, we re-design the adaptive controller from the preceding chapter to accommodate a tracking capability and eliminate the modular structure without losing the adaptive or convergence properties. It is shown that the tracking controller substantially enhances the maneuvering performance of the robot compared to the previous adaptive controller.

In Chapter 6, we demonstrate that, using the tools we have developed so far, and the iterative learning control algorithm, the insect-scale flapping-wing robot can mimic an aggressive maneuver seen in natural fliers. Perching on a vertical surface is chosen to represent a benchmark task that requires agility and accuracy of the control system. We demonstrate that, by the use of a simple magnetic attachment mechanism, and learning from previous mistakes, the robot can realize a pre-planed trajectory and successfully land on a vertical surface.

In Chapter 7, the model describing the translational and rotational dynamics of the robot is revisited. Recorded flight data gathered from previous experiments are processed and analyzed to provide empirical information on the dynamic and aerodynamic properties of the robot. Not only does this allow us to revise the existing dynamic model, but also to provide further insights into the dynamics and stability of the vehicle. Additionally, the obtained knowledge could be incorporated into the

flight controller design process for improved tracking performance.

We finally conclude with a discussion of the ongoing research and avenues to be addressed as steps towards the ultimate goal of developing a colony of autonomous robotic insects that can work in concert to accomplished assigned tasks. This encompasses the work required to transform the current 80-mg prototype operating in protective laboratory conditions to a sub-gram autonomous vehicle with integrated onboard sensors and flight avionics ready for operation in outdoor environments.

Nomenclature

The following nomenclature is assumed throughout this dissertation:

- In equations, bold letters indicate vectors.
- Given an *unknown parameter* α, its estimate is represented by $\hat{\alpha}$. The estimation error $\tilde{\alpha}$ is defined as $\tilde{\alpha} = \hat{\alpha} - \alpha$.
- Otherwise, $\hat{\cdot}$ represents a unit vector.
- High order derivatives are denoted by bracketed superscript, i.e., $\alpha^{(n)} = d^n\alpha/dt^n$.
- The variable s is reserved to represent a Laplace variable.

Chapter 2

Robot Model and Dynamics

In this chapter, we introduce the flapping-wing robotic insect prototype designed and fabricated in the Harvard Microrobotics Laboratory that the flight control experiments in this thesis are based upon. The physical properties and parameters of the robot are elaborated. In addition, simplified theoretical models explaining the mechanics, dynamics and the aerodynamic properties of the robot are outlined. These nominal models serve as a plant model for flight control purposes in later chapters.

2.1 Robot Description

The robot presented in this thesis (illustrated in figure 2.1) is an 80 mg flapping-wing robot fabricated using the *Smart Composite Microstructures* (SCM) process as detailed in [58, 81]. The robot's airframe is made of layers of carbon fiber, laminated under heat and pressure to form a rigid and lightweight composite, and laser machined. Resilient flexure joints are fabricated from polyimide film. The assembly pro-

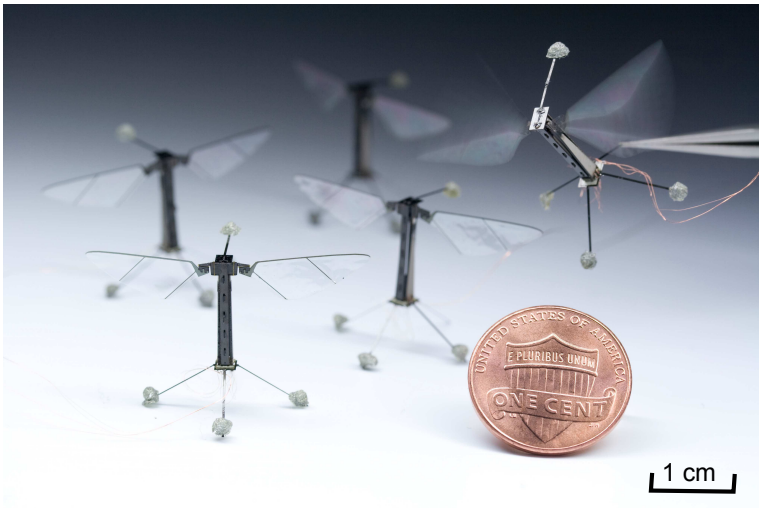

Figure 2.1: Five individual robotic insects of identical design are shown alongside a U.S. penny for scale, demonstrating that the manufacturing process facilitates repeatability and mass production.

cess takes inspiration from origami to create complex 3D structures and mechanisms by folding. Taking inspiration from *Diptera* (flies), the robot has two wings and is equipped with two piezoelectric bimorph actuators. Piezoelectric bimorph actuators are chosen over electromagnetic motors for the flight muscles due to their favorable scalability [93]. The addition of the second actuator distinguishes this robot from the earlier prototype in [69], allowing two wings to be driven independently, resembling the direct flight muscles found at the base of the wings in *Odonata* (dragonflies) [20]. This design also deviates from the robot presented in [28, 29], where two small control actuators were mounted at the base of each wing to independently fine tune the stroke amplitudes of the wings, similar to the thoracic topology of a Dipteran insect [20]. The physical parameters of the robot reported in [57] are reproduced in Table 2.1.

When a voltage is applied to the actuator, it induces motion at the tip of the

Table 2.1: Various physical parameters of the robotic insect.

Robot Properties		
Total mass	80	mg
Flight muscle mass	50	mg
Tracking marker mass	5	mg
Wire tether mass	0.2	$mg \cdot cm^{-1}$
Roll axis inertia	1.42	$g \cdot mm^2$
Pitch axis inertia	1.34	$g \cdot mm^2$
Yaw axis inertia	0.45	$g \cdot mm^2$
Reynolds number	<1200	
Flapping frequency	120	Hz
Flapping amplitude	110	degrees
Power consumption during hover	19	mW
Robot Geometry		
Height	14	mm
Body width	3.5	mm
Wing span	35	mm
Wing Properties		
Wing length	15	mm
Mean chord length	3.46	mm
Area	52	mm^2
Inertia (flapping axis)	45.3	$mg \cdot mm^2$
Mass	1	mg

actuator. In this robot, linear displacement of the actuator tip is amplified and converted into a rotational motion of the wing (described the angle Φ in figure 2.2) by a flexure-based spherical four-bar transmission, creating an actuator-transmission-wing system. The angle of attack of both wings is not directly controlled and relegated to a passive mechanism by the incorporation of compliant flexures at the wing hinges as shown in figure 2.2. The passive mechanism substantially simplifies the fabrication and control requirements. Lift force is then produced as a result of the wing rotation along the angle Ψ. In operation, the flapping frequency is typically fixed at a value between $110-120$ Hz, near the system's resonant frequency to maximize the flapping stroke amplitude and minimize reactive power expended by wing inertia and the hinge stiffness. The robotic insect is capable of modulating the thrust force that is nominally aligned with the robot's vertical axis by altering its flapping amplitude and able to generate torques along its three body axes: roll, pitch, and yaw, using different flapping schemes as detailed in figure 2.3. Theoretically, this allows the robot to be controllable over the $SO(3)$ space. Similar to most insects, which lack the ability to generate lateral thrust without banking [83], lateral maneuvers can be achieved by re-orienting the body such that the net thrust vector takes on a lateral component as modeled in the literature [96, 36]. This leaves researchers the task of designing a controller that determines the required lift and torque to stabilize the robot's flight. More details on the robot design and torque generation schemes can be found in [58].

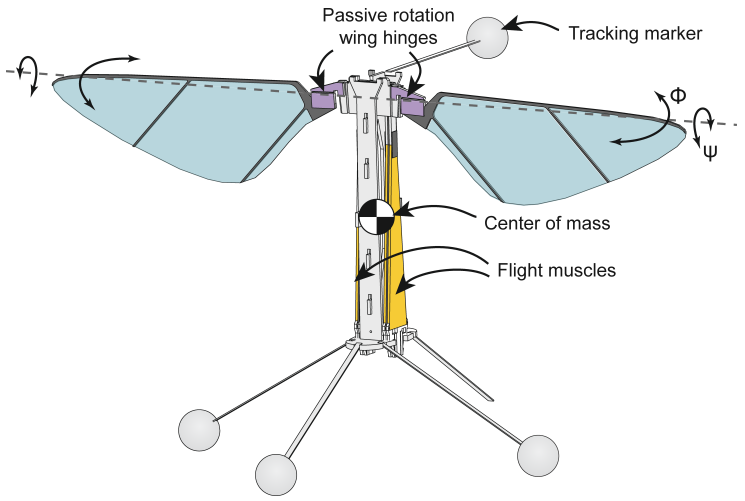

Figure 2.2: A schematic model highlights key components of the insect-scale robot, the rotational axes of the flapping strokes, and the passive rotation axes.

2.2 Commanded torques to wing trajectories

In [58, 69], it was shown that the thrust produced by the robotic insect is approximately a linear function of the actuator voltage amplitude. The robot was capable of producing thrust greater than 1.3 mN, or more than 1.5 times its own weight. Body torques on the order of one μNm can be achieved by using the three different flapping schemes illustrated in figure 2.3. Roll torque is generated by differentially changing the stroke amplitude of the two wings. Pitch torque is generated by shifting the location of the mid-stroke planes forward or backward. Adding a second harmonic signal into the flapping trajectory results in a difference in stroke velocity on the forward and backward strokes. This influences drag forces and alters the wing angle of attack. When applied to both wings in the opposite directions, a yaw torque is generated. These torque generation modes are greatly simplified in comparison to observed insect wing kinematics [83]. Nevertheless, they are often chosen for simulations and

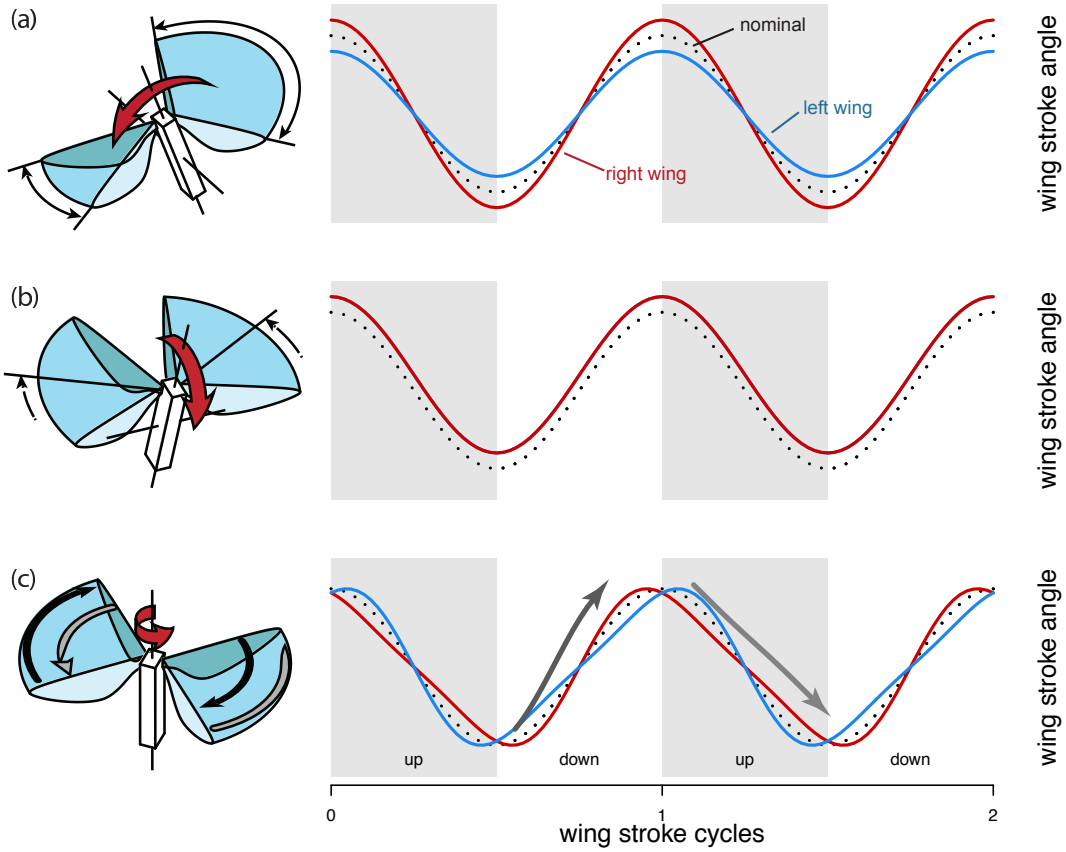

Figure 2.3: A robotic fly with a pair of independently actuated wings enables precise control of torques about three orthogonal axes. (a) Roll torque is generated by flapping one wing with a larger stroke amplitude than the other, inducing differential thrust forces. (b) Pitch torque is generated by moving the mean stroke angle of both wings forward or backward to offset the thrust vector away from the center of mass. (c) To generate yaw torques, the robot influences wing drag forces by cyclically modulating the stroke velocity in a "split-cycle" scheme [65].

dynamically-scaled robot experiments [19, 18, 65].

For control purposes, first the model of the system–a map or a transfer function between input signals and the resultant thrust or torques–has to be identified. The key challenge in obtaining such a model is the lack of a viable multi-axis force/torque sensor. In [58], a custom dual-axis force-torque capacitive sensor similar to the design in [26] was used to measure a single axis of torque and a single force perpendicular to the torque axis. This sensor, therefore, cannot determine the coupling between torques along different axes. Furthermore, despite over a decade of progress in micro-manufacturing, there still exists considerable variation between robots. Additionally, the process of mounting the robot on the sensor is challenging and possibly destructive, making it impractical to characterize all robots prior to flight experiments.

As a consequence, a more theoretical approach is taken. Quasi-steady analyses are often employed to capture the aerodynamic forces in insect flight [83, 5]. Taking a similar approach, in [88], the blade-element method was used to provide estimates of the aerodynamic forces and moments. This study is particularly suitable for our robot as it was carried out using wings operating at the same scale and Reynolds number as our robot. Moreover, the passive wing hinges present on the robot in this work were also used in the experiments and modeling in [88].

The model in [88] is used to compute the estimates of the resultant thrust and body torques using the flapping schemes shown in figure 2.3. Based on this, we constructed a theoretical approximation of the time-averaged thrust and torques as a function of wing trajectory. The findings suggests that, for constant frequency and over a small range of inputs, aerodynamic thrust can be approximated as a linear

function of the flapping amplitude, irrespective of other torque input parameters. This is consistent with the empirical evidence in [69]. Similarly, torques about the body axes are approximately linear functions of their respective input parameters as previously reported in [58]. The model further predicts minimal coupling between the three torque generation modes (in agreement with findings and assumptions in related work [19, 18]), and suggests that the torques are also dependent on the flapping amplitude. These can be summarized into a set of equations as followings:

$$\begin{aligned} \Gamma &= a_t \Phi_0 - b_t \\ \tau_r &= (a_r \Phi_0 - b_r)\, \delta_r \\ \tau_p &= (a_p \Phi_0 - b_p)\, \delta_p \\ \tau_y &= (a_y \Phi_0 - b_y)\, \eta, \end{aligned} \tag{2.1}$$

where Γ denotes the thrust, τ_i's represent roll, pitch, and yaw torques, Φ_0 is the flapping amplitude, δ_r is the differential stroke angle, δ_p is the shift in mean stroke angle, η is a relative proportion of a second-harmonic signal used for generating imbalanced drag forces, and the a_i and b_i terms are constants resulting from the linearization. The results from this equation are illustrated in figure 2.4, where predicted torques are plotted as functions of various wing stroke parameters. It suggests that, for example, the magnitude of pitch torque is more severely affected by the flapping amplitude than that of the roll torque. All in all, the predictions from equation (2.1) have to be treated with care as they only represent the nominal theoretical values that may vary notably from robot to robot. Any possible unknown discrepancies have to be

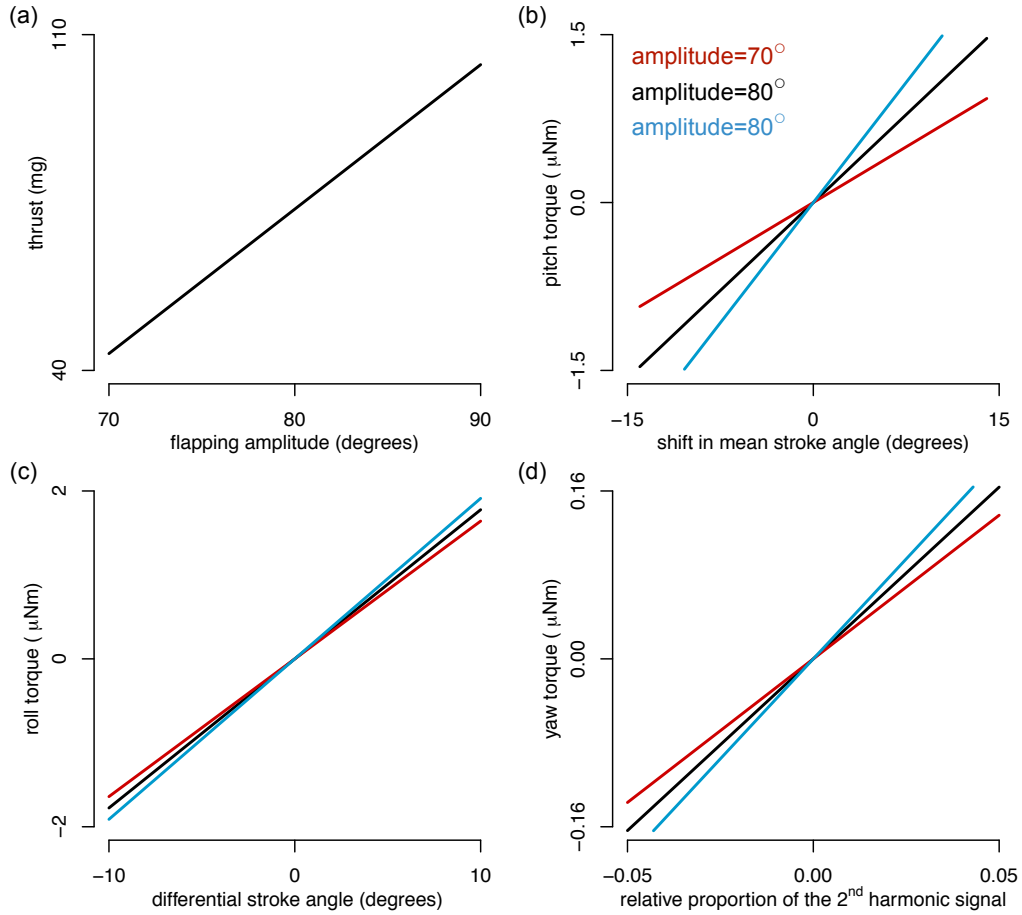

Figure 2.4: The generated force and torques as predicted by the aerodynamic model [88]. (a) Thrust (plotted in mg) approximately depends only on the peak-to-peak flapping amplitude. (b) Pitch torque is a linear function of the shift in the mean stroke angle, but is also dependent on the flapping amplitude. (c) Similarly, roll torque depends on the differential stroke angle. Its dependence on the flapping amplitude is less pronounced. (d) Yaw torque is more difficult to generate, but the moment of inertia is also smaller along the yaw axis. It can be approximated as a linear function of the amount of the 2^{nd} harmonic component in the driving signal.

robustly handled or compensated by the flight controller.

2.3 Linearized System Model

Once we have obtained the required wing trajectory for the desired thrust and torques, the corresponding actuator drive signals are calculated by approximating the actuator-transmission-wing system as a second order linear system [27]. To elaborate, the piezoelectric actuator also acts as a spring-like component to store elastic energy while driving the wing inertia, inducing an oscillatory behavior akin to the coupled muscle-thorax-wing system in flies [20]. As a consequence, a shift in the mean stroke angle, for example, translates to a DC offset in the drive signal. The model enables us to calculate the voltage amplitudes and offsets required to generate thrust to stay aloft and torques for control. Based on the predictions, one could also ensure that the total voltage required does not exceed the maximum actuator voltage.

In the following chapters, we assume that the torques can be directly commanded by the controller, and the delay in the actuation can be neglected (At 120 Hz, we anticipate that the robot would take a few flapping strokes (one flapping stroke equates to ≈ 8 ms) to realize the torque commands. These assumptions are also common in the MAVs community [53, 59, 52].

2.4 Time-Averaged Dynamic Equations

Owing to the relatively small inertia of the wings (relative to the body) and rapid but low-amplitude motion of the actuators, for the time scales of interest, these small

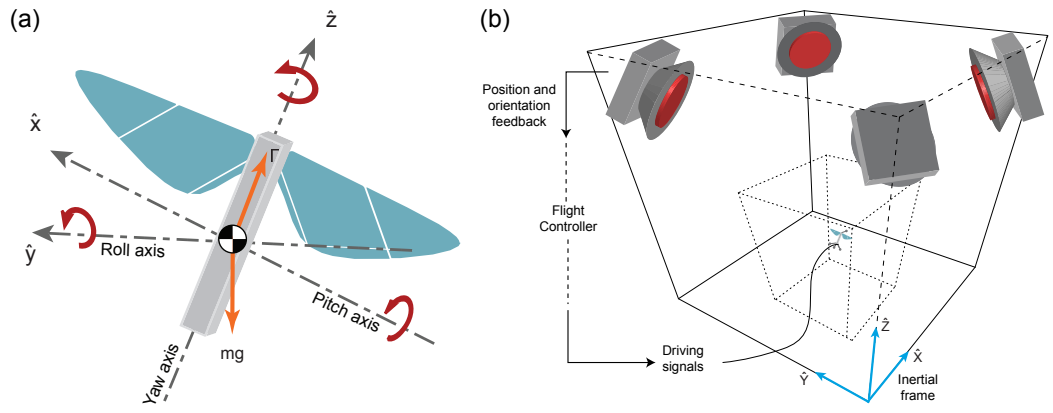

Figure 2.5: (a) Body axes definitions. (b) Four to eight infrared, motion tracking cameras observe the positions of retroreflective markers attached to the robot in order to estimate its position and orientation in space with low latency. Position estimates are transmitted to the controller computer, which computes the control signals and sends them to the robot via a wire tether.

oscillations are neglected and only stroke-averaged dynamics are considered. Stroke-averaged models were found to be sufficiently accurate to capture the dynamics of insects and robots of similar scales [18, 69]. Herein, the robotic insect is regarded as a rigid body in three-dimensional space, based on the standard aircraft model–the approach often taken in modeling the flight dynamics of flapping-wing MAVs [67, 75]. In the body attached coordinates, the roll, pitch, and yaw axes are aligned with the $\hat{x}$, $\hat{y}$, and $\hat{z}$ axes as presented in figure 2.5(a).

2.4.1 Rotational Dynamics

Due to symmetry, it is reasonable to assume that the cross terms in the moment of inertia matrix $\mathbf{J}$ are negligible. The orientation between the body frame and the inertial frame is defined by the rotation matrix R, which is rotating at an angular velocity ω with respect to the body frame. As a result, the attitude dynamics can be

described by the Euler equation

$$\mathbf{J}\dot{\omega} \;=\; \sum \tau - (\omega \times \mathbf{J}\omega)\,, \tag{2.2}$$

where $\Sigma\tau_i$ is the total torque acting on the vehicle. The total torque acting on the robot is the sum of the baseline torque generated from flapping the wings according to the schemes illustrated in figure 2.3 as a result of the control command and the additional damping of the aerodynamic effects that arise in free flight [75, 82, 18], which may be considered insignificant while the robot is stationary during hovering.

Given orientation feedback, one can construct a 3×3 rotation matrix (R) relating the body frame to the inertial frame. Note that since the elements of R can also be represented using $\hat{x}, \hat{y}$, and $\hat{z}$ as $R = \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \end{bmatrix}$. It follows that the angular velocity can be written as a function of the body axes and its time derivative, or the rotation matrix and its time derivative as

$$\begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix} \;=\; \begin{bmatrix} \hat{z}\cdot\dot{\hat{y}} & \hat{x}\cdot\dot{\hat{z}} & \hat{y}\cdot\dot{\hat{x}} \end{bmatrix} \quad . \tag{2.3}$$

The time derivative of the rotation matrix $\dot{R}$ is given as $\dot{R} = R\,[\omega\times]$, where the map $[(\cdot)\times] : \mathbb{R}^3 \mapsto \mathrm{SO}\,(3)$ is defined such that $[\mathbf{a}\times]\,\mathbf{b} = \mathbf{a} \times \mathbf{b}$ for all $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$. An alternative representation of $\dot{R}$ can be written as:

$$\begin{aligned} \dot{R} \;&=\; \begin{bmatrix} \dot{\hat{x}} & \dot{\hat{y}} & \dot{\hat{z}} \end{bmatrix} \\ &=\; \begin{bmatrix} \omega_z\hat{y} - \omega_y\hat{z} & \omega_x\hat{z} - \omega_z\hat{x} & \omega_y\hat{x} - \omega_x\hat{y} \end{bmatrix}. \end{aligned} \tag{2.4}$$

2.4.2 Translational Dynamics

The translational dynamics of the robot depends on the orientation of the robot. In other words, the normalized thrust Γ (which has a dimension of acceleration, not force) is nominally aligned with the $\hat{z}$ axis of the robot. It follows that we can write the equation of motion of the robot in the inertial frame as

$$m \begin{bmatrix} \ddot{X} & \ddot{Y} & \ddot{Z} \end{bmatrix}^T = m\mathbf{g} + m\Gamma\hat{z} + F_{aero}, \tag{2.5}$$

where m denotes the mass of the robot, X, Y, and Z are position of the robot in the inertial frame, and F_{aero} represents additional aerodynamic forces that arise in free flight. Such effects, including unsteady flow, are difficult to accurately capture using simple models that are suitable for real-time control purposes. These damping effects are often neglected in the control problems of MAVs [44, 53, 59]. In the study of insect flight, however, it has been shown that they could be approximately captured as linear terms in the rotational dynamics and the translational dynamics [24, 75]. While these additional terms may be negligible in controlled hovering flight, they could become significant in more aggressive flight. More details on these aerodynamic effects are given in Chapter 7.

2.5 Experimental Setup

Flight control experiments are performed in a flight arena equipped with four to eight motion capture *VICON* cameras (see figure 2.5b), providing a tracking volume of $0.3 \times 0.3 \times 0.3$ m. The system provides position and orientation feedback by tracking

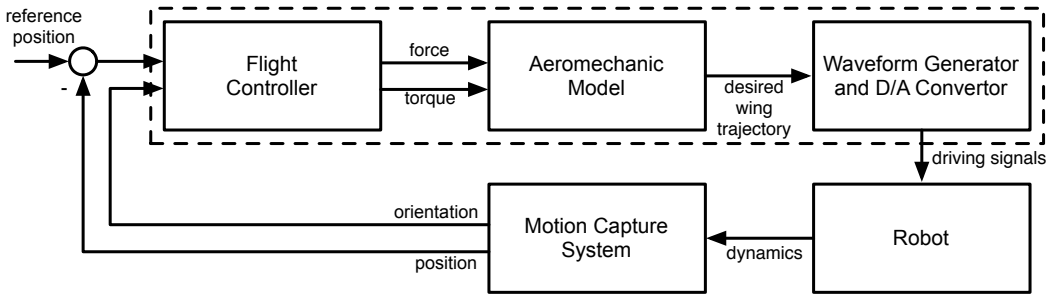

Figure 2.6: Block diagram illustrating the control architecture. The flight controller determines the desired values of thrust and torque to be generated by the robot. The control forces and torques are converted first to wing trajectories and then to input signals for the robotic fly. The body dynamics of the robot are measured using a 3D motion tracking system.

the position of four retroreflective markers at a rate of 500 Hz. Orientation feedback is given in the form of Euler angles that can immediately be converted into a rotation matrix. Computation for control is carried out on external computers using an xPC Target system (*Mathworks*), which operates at 10 kHz for both input sampling and output signal generation. High-voltage amplifier delivers power to the robot via a 0.6 m long bundle of four 51-gauge copper wires. A schematic diagram illustrating the experimental setup is shown in figure 2.6. The latency of the complete experimental setup was found to be approximately 8 ms–around wing beat (See Appendix A), comparable to the measured neural delay time of fruit flies [75].

Chapter 3

Nonlinear Flight Controller

Flies are among the most agile flying creatures on Earth. Their superlative maneuverability is a result of sophisticated sensory-motor systems working against intrinsic flight instabilities. To mimic this aerial prowess in a similarly sized robot with limited information about the robot's dynamics requires a nonlinear control solution that is capable of stabilizing the robot over a large region of attraction. To this end, we demonstrated tethered but unconstrained stable hovering and basic controlled flight maneuvers. The result validates a sufficient suite of innovations for achieving artificial, insect-like flight.

3.1 Introduction

Recent development of flight-capable insect-scale flapping-wing robots that are capable of independently modulating roll, pitch, and yaw torques as well as producing enough lift force [29, 28, 58] provides researchers with an opportunity to study and

expand their understanding of the flight control in insects and millimeter-scale MAVs. Flight control of vehicles at such scale could be markedly different from that of larger flying vehicles due to a number of reasons: the intrinsic instability of insect flight [67, 8], the difference between the aerodynamics of flapping-wing flight and fixed-wing flight, and the relative importance of disturbances, to name but a few.

To date, there have been a number of theoretical studies on insect flight control using various techniques [15, 66, 19, 67], including linear control, optimal control, etc. While these proposed controllers offer plausible solutions, they are devised based on different assumptions regarding the dynamics and modes of torque generation which vary dramatically across a range of vehicles. Moreover, practical considerations such as unmodeled dynamic effects, disturbances, and sensing are often ignored. This motivates us to design a nonlinear flight controller that specifically takes into consideration the information regarding dynamics of the robot and the limitation of our experimental system outlined in the preceding chapter.

In section 3.2, we present the derivation and stability analysis of the proposed nonlinear flight controller and outline a method for compensating for imperfections from the fabrication process. Results from unconstrained flight experiments are shown in section 3.3, followed by conclusion and discussion.

3.2 Controller design

The body dynamics of the millimeter-scale robot are fast—based on the measured torque generation, the robot should be able to perform a 90° turns in less than 30 ms [58]. To stabilize such dynamics, the controller must be considerably faster. We use

a 10 kHz controller operating frequency — on the order of 100 times faster than the nominal 120 Hz flapping frequency to obtain sufficiently smooth signals.

Complicating the control problem, we observe considerable and unpredictable variations from robot to robot due to small manufacturing differences that are difficult to characterize. To date, we have not found a commercially-available sensor with a suitable range (≈1-10 μNm) and resolution (≈10 nNm), and possessing multiple, simultaneous measurement axes. A custom, dual-axis force-torque sensor capable of measuring a single axis of torque and a single force perpendicular to the torque axis of suitable range was demonstrated and used to measure thrust force magnitudes from the robot [58]. Because there is no closed-form solution to the Navier-Strokes equations for flapping-wing flight, we must use approximations of the various features of the aeromechanical system to create a simple plant model as presented in Chapter 2. The rigid body dynamics in three-dimensional space are also markedly nonlinear. The combination of these uncertainties and the rapid dynamics of the system presents a challenging control problem.

The approach we are using here initially is to focus on achieving stable hovering behavior, dividing the controller into three modules: attitude, lateral position, and altitude controllers as described below. The overall architecture of the flight controller is schematically shown in figure 3.1.

3.2.1 Attitude Controller

For the purposes of hovering, it is not necessary for the flapping-wing robot to maintain a specific yaw angle heading because the robot can roll or pitch to move

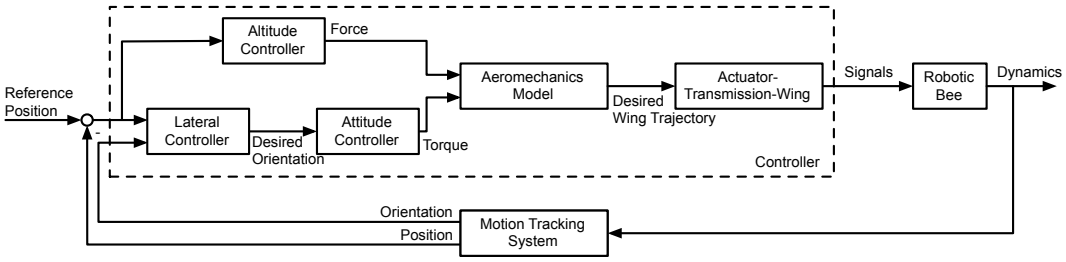

Figure 3.1: Block diagram illustrating the control architecture. The controller contains three modules to control attitude, lateral position, and altitude. The calculated control forces and torques are converted first to wing trajectories and then to input signals for the robotic fly. The body dynamics of the robot are measured using a 3D motion tracking system.

laterally in any direction. However, our attitude controller applies a damping effect to counteract rotation because otherwise, the slight fabrication asymmetries between the two wings results in rapid spinning behavior that destabilizes the robot.

A few assumptions regarding the dynamics of the robot are made to simplify the attitude controller design. First, the robot is treated as a rigid body, neglecting the inertial effects of the mass of the wings. Second, because we want stationary hovering, we assume the wind strength and translational and rotational velocities are near zero, so that we can neglect aerodynamic forces and torques arising from translational and rotational motions of the robot.

Accordingly, we can model the control problem as applying forces and torques to stabilize a simple rigid body under the influence of gravity alone. The attitude controller stabilizes the body in 3-D space using an energy-based Lyapunov function, motivated by [54]. The function takes the form:

$$V_0 = k_p \left(1 - \cos \Phi\right) + \frac{1}{2} \omega^T I \omega,$$

where k_p is a positive scalar, I is the rigid body inertia matrix in the body coordinate frame, ω is the angular velocity, and Φ is defined as the angle between the current body axis orientation ($\hat{z}$) and the desired body axis orientation ($\hat{z}_d$) such that

$$\hat{z} \cdot \hat{z}_d = \cos\Phi.$$

The proposed Lyapunov function is a positive definite function and is zero only when the robot is oriented in the desired orientation and stationary. Assuming $\hat{z}_d$ is constant or slowly varying, the time derivative of the function is given by

$$\dot{V}_0 = k_p \omega^T \left(\begin{bmatrix} R_{12} & R_{22} & R_{32} \\ -R_{11} & -R_{21} & -R_{31} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \\ \hat{z}_{d3} \end{bmatrix} \right) + \omega^T \tau,$$

where we have used equations (2.2) and (2.4), R_{ij} represents an element in the rotation matrix describing the current orientation of the robot, $\hat{z}_{di}$ is an i^{th} element of the vector $\hat{z}_d$, and τ is the command torque vector to be generated by the robot. It follows that, for a positive constant k_v, the following control law

$$\tau = -k_p \left(\begin{bmatrix} R_{12} & R_{22} & R_{32} \\ -R_{11} & -R_{21} & -R_{31} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \\ \hat{z}_{d3} \end{bmatrix} \right) - k_v \omega$$

yields

$$\dot{V} = -k_v \omega^T \omega \leq 0.$$

According to the invariant set theorem, the system is globally asymptotically stable with zero attitude error and zero angular velocity under the assumptions given above.

In practice, however, the angular velocity feedback is not directly available from the motion capture system. To ensure that the stability condition is retained, the Lyapunov function is modified with an additional term:

$$\begin{aligned} V &= V_0 + \frac{1}{2} k_v \chi^T \chi \\ &= k_p \left(1 - \cos \Phi\right) + \frac{1}{2} \omega^T I \omega + \frac{1}{2} k_v \chi^T \chi, \end{aligned}$$

where the dynamics of χ are described by

$$\chi = \frac{s}{s + \lambda} e.$$

Here s is a Laplace variable, λ is a positive constant, and e is a 3×1 vector containing Euler angles representing the orientation of the robot. There exists a matrix $E(e)$ that maps $\dot{e}$ to the angular velocity as

$$\dot{e} = E(e)\omega.$$

By using the control law

$$\tau = -k_p \left(\begin{bmatrix} R_{12} & R_{22} & R_{32} \\ -R_{11} & -R_{21} & -R_{31} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \\ \hat{z}_{d3} \end{bmatrix} \right) - k_v E^T(e) \chi,$$

the time derivative of the Lyapunov function is given by

$$\begin{aligned} \dot{V} &= -\lambda k_v \chi^T \chi - k_v \omega^T E^T \chi + k_v \chi^T \dot{e} \\ &= -\lambda k_v \chi^T \chi. \end{aligned}$$

Subsequently, the global asymptotic stability is achieved without directly using the angular velocity feedback. The resultant control law has a structure similar to a PD controller–the first term corresponds to a proportional error and the second term can be regarded as a derivative error. The values of k_p, k_v, and λ were experimentally tuned.

3.2.2 Lateral Controller

To navigate the robotic fly to the setpoint lateral positions, the outer lateral controller determines a desired body attitude of the robot ($\hat{z}_d$). By commanding tilt angles, the lateral controller causes the thrust vector to take on a lateral component. This desired attitude orientation is inputted to the attitude controller, which acts to regulate attitude to this new orientation. The magnitude of the lateral force is a function of the deviation angle between the instantaneous body axis and the vertical axis. This controller module is a proportional-derivative controller of the form:

$$\begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \end{bmatrix} = -k_{pl} \begin{bmatrix} X - X_d \\ Y - Y_d \end{bmatrix} - k_{vl} \begin{bmatrix} \dot{X} - \dot{X}_d \\ \dot{Y} - \dot{Y}_d \end{bmatrix},$$

where X, Y, X_d, Y_d are positions and desired positions, k_{pl} and k_{vl} are controller gains. $\hat{z}_{d3}$ is calculated such that the magnitude of the vector $\hat{z}_d$ is unity. Additionally, a saturation scheme was also implemented to ensure that the setpoint attitude does not deviate by more than 30° from the upright orientation.

3.2.3 Altitude Controller

The altitude controller determines the amount of propulsive force required to reach an altitude setpoint. It is a proportional-derivative controller, with a feedforward term to account for the gravity, and assumes a one-dimensional, upright oriented system.

However, since we are not executing rapid flight maneuvers that may require extreme body attitude angles (over 30° of tilt) as presented in Chapter 6, the altitude controller is deliberately designed to ignore the attitude of the robotic insect. In other words, it assumes the system is always upright and adjusts the propulsive force magnitude to modulate vertical lift force and consequently vertical position. The approximation, which amounts to a linearization about an operating point at hover, holds because the robotic bee tilts less than 30° to generate lateral forces for lateral position control. This deviation of the propulsive force vector only marginally disrupts the vertical lift force component.

Assuming a nearly upright orientation offers two beneficial consequences. First, it avoids commanding an increase in thrust when the robot tilts from the upright orientation, leaving more voltage adjustment range to the more critical attitude controller (voltage range is limited by the voltage rail at 300V). Second, it reduces the amount of lateral drift that would arise if the robot increases thrust to compensate for off-axis

tilting away from the upright orientation.

3.2.4 Compensation for fabrication-based torque biases

Despite attempts identify input signals that minimize generated body torques, practically it is impossible to completely eliminate such bias torques. These bias torques are the result of inevitable asymmetries due to manufacturing variations. The asymmetries offset the net propulsive thrust vector from the robot's center of mass, causing undesired residual torques about the body. This open-loop "trimming" procedure provides a rough estimate of the zero-body-torque state and is sufficient information to achieve short, marginally stable hovering flights. However, we find that the robotic fly steadily drifts laterally out of the control volume due to estimation inaccuracy. To achieve stable, stationary hovering, we compensate for the error by adding an integral term in the attitude controller to achieve zero-body-torque operation at steady-state. A secondary issue is that the zero-body-torque state does not necessarily coincide with the propulsive force vector of the robot oriented vertically, contributing additional lateral drift. We implement an algorithm in the lateral controller to evaluate the direction of the misaligned propulsive thrust vector and redefine the zero-body-torque state to remove the misalignment. This is carried out by considering a simplified model of the lateral dynamics of the robot, assuming steady hover (i.e. when lift thrust balances the gravitational force):

$$mg \begin{bmatrix} R_{13} \\ \\ R_{23} \end{bmatrix} + mg \begin{bmatrix} r_X \\ \\ r_Y \end{bmatrix} = m \begin{bmatrix} \ddot{X} \\ \\ \ddot{Y} \end{bmatrix} + b \begin{bmatrix} \dot{X} \\ \\ \dot{Y} \end{bmatrix},$$

where g is the gravitational constant, b is the aerodynamic drag coefficient, R_{13} and R_{23} are elements of the rotation matrix, and r_X and r_Y represent the misalignment of the body axes along the inertial coordinate frames. Here r_X and r_Y are determined using filtered derivatives on X and Y and projected back onto the robot's body frame. They are continuously estimated using a low-pass filter. A revised rotational matrix corresponding to the observed dynamics is then obtained.

3.3 Flight Experiments

Prior to the flight experiments, the robot underwent a series of tests on a static setup for inspections of the flapping frequency and amplitude. This procedure allows us to verify that the robot possesses a resonant frequency and large symmetrical flapping strokes that satisfy the requirements for lift generation and balanced torque production.

Flight experiments were performed in a laboratory environment as described in Section 2.5. Open-loop experiments were carried out prior to the closed-loop flights in order to identify input signals that minimize bias torques and approximate an operating state where there is zero body torque.

In flight tests, the robotic fly demonstrated stable hovering about a fixed point with position errors on the order of one body length around the target position sustaining flights for longer than 20 seconds without ever approaching a crash. Images showing the robot in an example of a 10-second hovering flight are shown in figure 3.2. The robot lifted off and reached the altitude setpoint at ≈ 8.5 cm in approximately 0.6 s and stayed within 5 cm of the setpoint throughout the flight, after which the

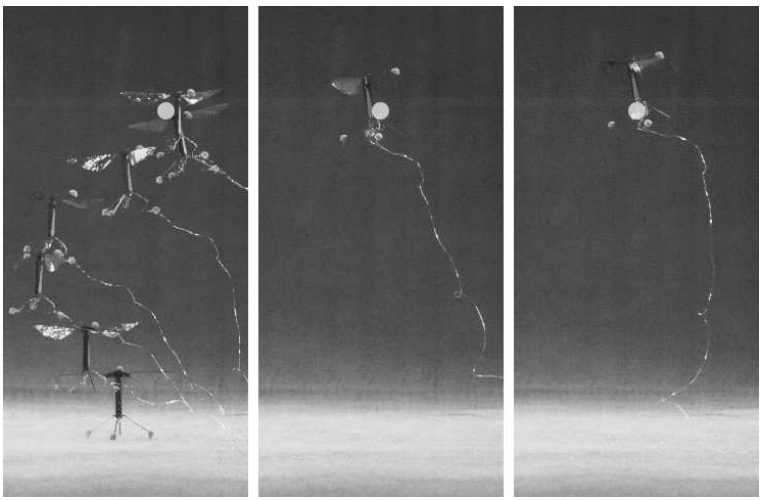

Figure 3.2: Frames from a recorded flight video at various times during the robot's flight. (left) Strobed positions at 0.1-s intervals. The white dot indicates the desired hovering location.

controller was switched off. Reconstruction of the flight trajectory is shown in figure 3.3. Additionally, the robot also demonstrated lateral flight maneuvers, alternating between two fixed points in space by a switch of the target lateral position. Figure 3.4 shows the robot traversing 20 cm laterally in less than one second. It can be seen that the lateral controller commanded the position of the robot by modulating the body orientation of the robot as the desired body axis. The attitude controller then realized the body orientation and simultaneously stabilized the robot.

3.4 Conclusion and Discussion

Motivated by the agile dynamics and the lack of system information, in this chapter we designed a nonlinear flight controller suite using an energy-based Lyapunov function with a large region of attraction. This is distinguishable from linearized

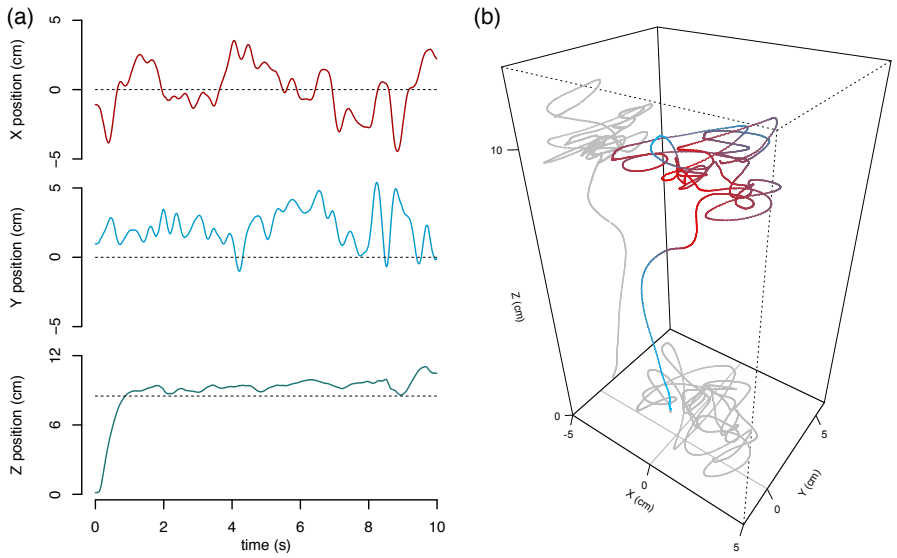

Figure 3.3: (a) Recorded trajectory of the robot from a 10-second hovering flight projected onto the inertial coordinate frame. Dashed lines are setpoint positions. (b) The 3D reconstruction of the flight trajectory. Line color gradient indicates distance from the target point, with red indicating closer proximity.

approaches where stability is only guaranteed in a small domain. We have verified that the proposed method successfully stabilized the flapping wing robot in hovering flights, marking the first demonstration of unconstrained flight of a flapping-wing vehicle at this scale.

While difficult to quantify, we believe the tether wires have a minor destabilizing effect on the dynamics of the robot. Due to the widely varying conformation of the wires as the robot performs flight maneuvers, it is unlikely that the tether wires provided any consistent stabilizing effects. We treat the effect of the tether as a disturbance that is compensated for in the open-loop trimming experiments and by the controller. More thorough analysis on the effects of the tether can be found in Appendix B.

Because of its scale and ability to perform stable, controlled flight, the flapping-

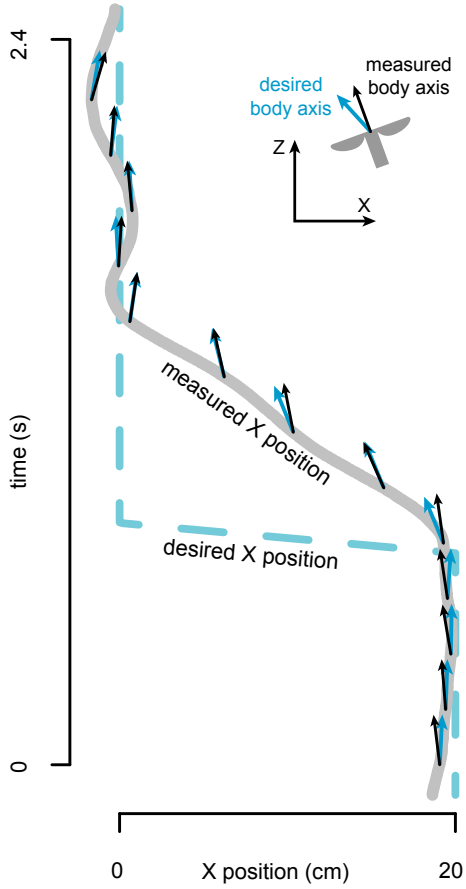

Figure 3.4: The robotic fly executes a lateral maneuver. During the maneuver, the altitude remains roughly constant. With time plotted on the vertical axis, the orientation of the robot's body axis, as projected onto the x-z plane, tilts to generate lateral forces in response to a change in the desired lateral position (dashed line).

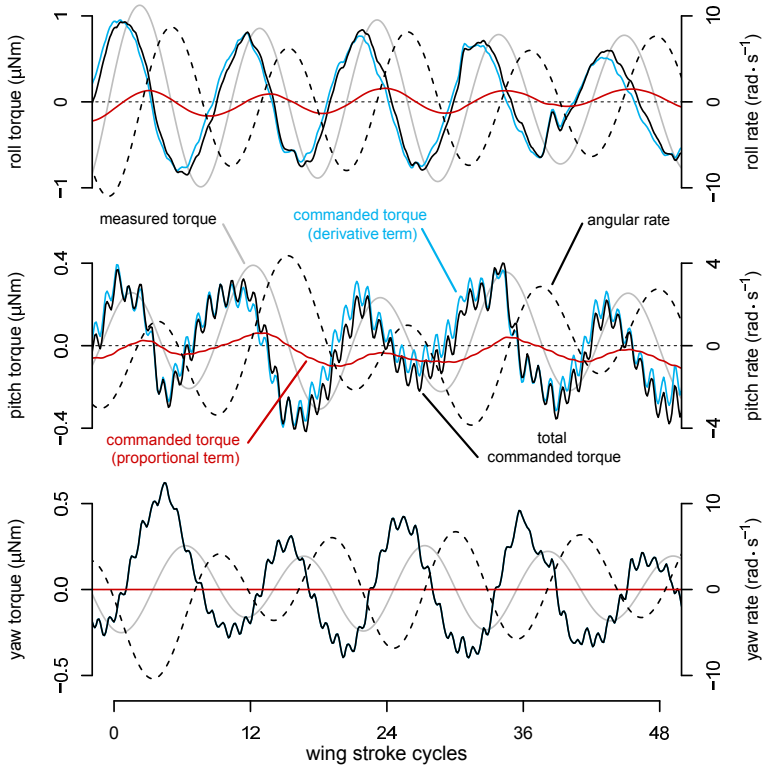

Figure 3.5: Commanded torque magnitudes during an attitude-stabilized flight indicate that stabilization torques are dominated by the derivative term of the controller. In the yaw torque plot, the proportional term does not contribute because, by design, only yaw rate is controlled. In all cases, the measured torque follows the commanded torque with a constant delay due to sensor latency and robot thoracic mechanics. Angular rate and measured torque (estimated from angular rate and the inertia of the robot) was low-pass filtered with a cutoff frequency of 60 Hz in order to reduce noise. The robot's flapping frequency is 120 Hz, and its effect can be observed as a higher-frequency oscillation superimposed on the commanded torque signals.

wing robot provides an alternative method for studying insect-scale, flapping-wing flight mechanics and flight control. For example, we studied the role of the flight controller on the attitude dynamics of the robot by turning off the lateral position control to emphasize torque commands from attitude stabilization. Our flight data suggest that the robot's attitude-stabilization torques are highly dependent on information about angular velocities (figure 3.5). The commanded torque counters the measured angular rate with a delay of approximately two wing-stroke cycles. This coincides with the biological observation that haltere-mediated feedback is rate-dependent in *Drosophila* [17] and the similar finding in theoretical models of other flying insects [8]. Additionally, flapping-wing flight experiences movement-based forces and torques that may be difficult to simulate in dynamically scaled fluid mechanics models. These dynamics, such as nonlinearities and cross-coupling of different degrees of freedom that arise during complicated flight maneuvers, could be measured with model fitting [42] or onboard sensors.

Chapter 4

Adaptive Control

Challenges for controlled flight of a robotic insect are due to the inherent instability of the system, complex fluid-structure interactions, and the general lack of a complete system model. In this chapter, we propose theoretical models of the system based on the limited information available from the preceding chapters and a comprehensive flight controller. The modular flight controller is derived from Lyapunov function candidates with proven stability over a large region of attraction. Moreover, it comprises adaptive components that are capable of coping with uncertainties in the system from manufacturing imperfections. We have demonstrated that the proposed methods enable the robot to achieve sustained hovering flights with relatively small errors compared to a non-adaptive approach presented in the previous chapter. Simple lateral maneuvers and vertical takeoff and landing flights are also shown to illustrate the fidelity of the flight controller. The analysis suggests that the adaptive scheme is crucial in order to achieve millimeter-scale precision in position as observed in insect flights in nature.

4.1 Introduction

In Chapter 3, controlled flight of a millimeter-scale flapping-wing robot was first empirically demonstrated. From a controls perspective, flapping-wing MAVs [13, 48] bear some resemblance to helicopters and quadrotors. Both types of aircraft are typically underactuated with four inputs and six rigid body degrees of freedom. While flying insects are able to perform extraordinary aerodynamic feats, small quadrotors are also known to possess great maneuverability [60, 63]. Given the extensive research on controlling quadrotors using various techniques [63, 60, 49], similar strategies may also be suitably applied to the emerging flapping-wing MAV prototypes.

In addition to swift dynamics, a primary challenge in controlling the robotic insect is due to the lack of comprehensive knowledge of the system and the variation in system properties owing to complex aerodynamics and manufacturing imperfections. Empirical characterization and system identification are not currently feasible since a multi-axis force/torque sensor with appropriate range and resolution for the robots of interest does not exist. To compensate, in Chapter 2, predictions of the system's characteristics were made based on theoretical models [88, 27]. In order to achieve sustained flight, it is necessary to account for uncertain parameters arising from manufacturing errors (e.g. torque offsets); this cannot be done by modeling alone. One possible approach to account for model uncertainties is to use an adaptive controller.

The controllers used in the Chapter 3 are not inherently adaptive. Instead, an integral part is added to deal with parameter uncertainty. It is conceivable that the use of adaptive controllers with proven convergence properties could improve flight performance. Additionally, the results allow us to gain further insights into the flight

dynamics of the vehicle and obtain more realistic models for control purposes. In this chapter we revisit the problem of controlling the robotic insect by employing an adaptive approach. The flight controller has been designed based on proposed Lyapunov functions using sliding mode control techniques. The control laws and adaptive laws are derived such that the stability can be guaranteed in a Lyapunov sense. The use of adaptive sliding mode control is not novel in MAV applications, nevertheless, earlier work often relies on a small-angle assumption [94, 52]. In this chapter, the proposed controller possesses a large domain of attraction. Another major benefit is the reliability of the adaptive parts that allow us to efficiently obtain estimates of uncertain parameters. The performance of the proposed controller is tested in more than 20 hovering flights and compared to the results from the non-adaptive controller from Chapter 3. Demonstrations of simple maneuvers are also presented, together with a more thorough analysis of the results. The outcomes suggest the importance of the adaptive scheme and its potential roles in the flight control system of real insects.

4.2 Controller Design

Driven by the lack of both empirical measurements and an accurately identified model of the robot, we employed an adaptive controller in order to estimate unknown parameters. The overall flight controller is comprised of three subcontrollers: a lateral controller, an attitude controller, and an altitude controller. In comparison with Dipteran insects, in our robot the lateral controller has slower dynamics and can be associated with the optomotor control system in insects, whereas the attitude

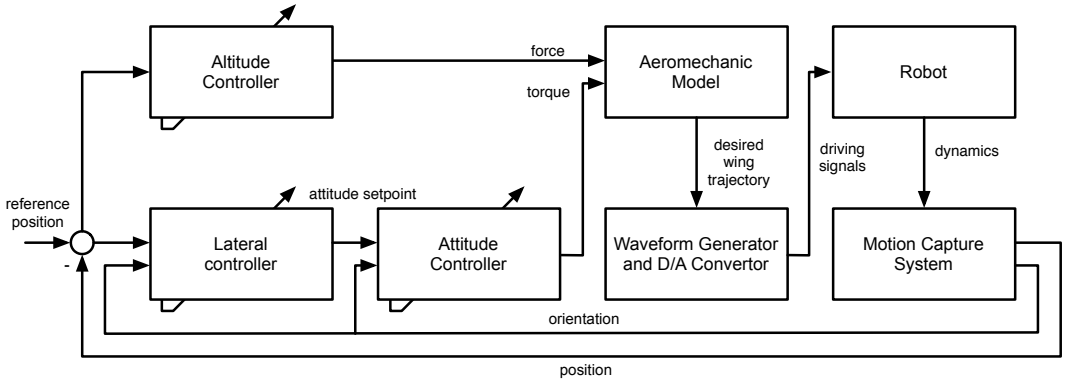

Figure 4.1: Block diagram explaining the experimental setup and control strategy. The cascaded lateral controller and attitude controller are operated in parallel with the altitude controller.

controller holds a role similar to the mechanosensory system for rapid feedback [83]. Here the lateral controller takes position feedback from a motion capture system and determines the desired orientation of the robot in order to maneuver the robot to a position setpoint. This desired orientation serves as the setpoint for the attitude controller that evaluates the required torques from the vehicle to achieve the desired attitude. In parallel, the altitude controller computes the suitable thrust force to maintain the robot at the desired height based on external position feedback. The block diagram representing these controllers is presented in figure 4.1. These controllers are considerably different from those in Chapter 3 as they employ the use of sliding mode control techniques [80] for adaptive purposes. Moreover, higher order models of lateral and altitude dynamics are implemented to reduce the oscillating behaviors seen in the results from Chapter 3.

4.2.1 Adaptive Attitude Controller

A consensus drawn from several stability studies indicates that, similar to insect flight, flapping-wing MAVs in hover are unstable without active control [67, 75]. Together with uncertainties due to an incomplete model of the vehicle and the requirement to vary the attitude setpoint for lateral maneuvers, it is necessary to design a robust controller that allows for significant excursions from the hovering state. As opposed to traditional linear controllers based on a linearization about hover, we employ Lyapunov's direct method to design a controller with a large domain of attraction. The attitude controller employed here is distinct from the one that demonstrated the first successful flights in the preceding chapter as it enables better tracking and adaptive ability for uncertain parameter estimates.

The goal of the attitude controller is to align the robot $\hat{z}$ axis with the desired attitude vector $\hat{z}_d$, without specifically controling the $\hat{x}$ and $\hat{y}$ axes. Such a strategy allows the robot to maneuver in the desired direction while relaxing control over the exact yaw orientation. In other words, the robot has no preference to roll or pitch, but a combination of them would be chosen so that the body $\hat{z}$ axis aligns with the desired attitude vector $\hat{z}_d$ with minimum effort.

Based on a sliding control approach [80], we begin by defining a sliding surface composed of an angular velocity vector ω and the attitude error $\mathbf{e}$,

$$\mathbf{s}_a = \omega + \Lambda \mathbf{e}, \tag{4.1}$$

where Λ is a positive diagonal gain matrix. The attitude error $\mathbf{e}$ is selected to corre-

spond to the amount of the deviation of $\hat{z}$ from $\hat{z}_d$,

$$\begin{aligned} \mathbf{e} &= \begin{bmatrix} \hat{y}\cdot\hat{z}_d & -\hat{x}\cdot\hat{z}_d & 0 \end{bmatrix}^T \\ &= \begin{bmatrix} R_{12} & R_{22} & R_{32} \\ -R_{11} & -R_{21} & -R_{31} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \\ \hat{z}_{d3} \end{bmatrix}. \end{aligned} \tag{4.2}$$

Note that the third element of the attitude error vector is zero, consistent with the decision not to control the exact yaw orientation. The composite variable $\mathbf{s}_a$ is zero when $\hat{z}$ aligns with $\hat{z}_d$ and the robot has no angular velocity, hence the controller would only command a yaw torque to neutralize the yaw rate and disregard the yaw orientation. This strategy is suitable for this particular vehicle, which was found to be able to produce relatively large roll and pitch torques, but its ability to produce yaw torque is limited.

Let $\hat{\alpha}$ be a vector containing the estimates of unknown parameters and $\tilde{\alpha}$ be the estimation error defined as $\tilde{\alpha} = \hat{\alpha} - \alpha$, we propose the following Lyapunov function candidate

$$V_a = \frac{1}{2}\mathbf{s}_a^T\mathbf{J}\mathbf{s}_a + \frac{1}{2}\tilde{\alpha}^T\Gamma^{-1}\tilde{\alpha}, \tag{4.3}$$

here Γ is a positive diagonal adaptive gain matrix. Assuming that the robot also produces some unknown constant torques $-\tau_o$ in addition to the commanded torque

τ_c by the controller, equation (2.2) can be rewritten as

$$\mathbf{J}\dot{\omega} = \tau_c - \tau_o - (\omega \times \mathbf{J}\omega)\,. \tag{4.4}$$

As a result, from equations (4.1), (4.3) and (4.4), the time derivative of the Lyapunov function is given by

$$\dot{V}_a = \mathbf{s}_a^T \left(\tau_c - \tau_o - (\omega \times \mathbf{J}\omega) + \mathbf{J}\Lambda\dot{\mathbf{e}}\right) + \dot{\tilde{\alpha}}^T \Gamma^{-1}\tilde{\alpha}. \tag{4.5}$$

Defining $\hat{\mathbf{J}}$ as the estimate of the diagonal inertia matrix, we propose the control law

$$\begin{aligned} \tau_c &= -K_a\mathbf{s}_a + \hat{\tau}_o - \left(\Lambda\mathbf{e} \times \hat{\mathbf{J}}\omega\right) - \hat{\mathbf{J}}\Lambda\dot{\mathbf{e}} && (4.6)\\ &= -K_a\mathbf{s}_a + Y\hat{\alpha}, && (4.7) \end{aligned}$$

where K is a positive diagonal gain matrix, $\hat{\tau}_o$ is an estimate of the unknown offset torque τ_o, and the matrix Y and the parameter estimate vector $\hat{\alpha}$ are

$$\begin{aligned} Y &= \left[\begin{array}{ccc|c} -\Lambda\dot{\mathbf{e}}_1 & \Lambda\dot{\mathbf{e}}_3\omega_y & -\Lambda\dot{\mathbf{e}}_2\omega_z & \\ -\Lambda\dot{\mathbf{e}}_3\omega_x & -\Lambda\dot{\mathbf{e}}_2 & \Lambda\dot{\mathbf{e}}_1\omega_z & I_{3\times 3} \\ \Lambda\dot{\mathbf{e}}_2\omega_x & -\Lambda\dot{\mathbf{e}}_1\omega_y & -\Lambda\dot{\mathbf{e}}_3 & \end{array}\right] \\ \hat{\mathbf{a}} &= \left[\begin{array}{cccccc} \hat{\mathbf{J}}_{xx} & \hat{\mathbf{J}}_{yy} & \hat{\mathbf{J}}_{zz} & \hat{\tau}_{o1} & \hat{\tau}_{o2} & \hat{\tau}_{o3} \end{array}\right]^T . \end{aligned} \tag{4.8}$$

Equation (4.5) then becomes

$$\dot{V}_a = -\mathbf{s}_a^T K_a \mathbf{s}_a + \mathbf{s}_a^T Y\tilde{\alpha} + \dot{\tilde{\alpha}}^T\Gamma^{-1}\tilde{\alpha}. \tag{4.9}$$

This suggests the adaptive law

$$\dot{\hat{\alpha}} = -\Gamma Y^T \mathbf{s}_a, \tag{4.10}$$

which renders the time derivative of the Lyapunov function to be negative definite,

$$\dot{V}_a = -\mathbf{s}_a^T K_a \mathbf{s}_a \leq 0. \tag{4.11}$$

According to the invariant set theorem, the system is theoretically almost globally asymptotically stable. That is, the composite variable and the estimation errors converge to zero. The exception occurs when the $\hat{z}$ axis points in the opposite direction to the desired attitude vector. Additionally, notice that no particular representation of rotation is used, hence no care needs to be taken to avoid a singularity or any ambiguity in the choice of representation.

The presented attitude controller has a few benefits over the controller employed in Chapter 3. For instance, it has better tracking ability, and the adaptive part takes into consideration the torque offset errors and uncertainty in the estimate of the inertia and makes corrections based on the feedback.

4.2.2 Adaptive Lateral Controller

The lateral controller is designed based on the dynamics described in equation (2.5). This controller relies on position feedback to compute the desired attitude vector that becomes a setpoint for the attitude controller. An adaptive part is incorporated in order to account for misalignment between the presumed thrust vector and the true orientation of the thrust vector. Moreover, the lateral controller assumes

that the response of the attitude controller can be described by a first order differential equation as shown in equation (4.12)–this consideration was not present in the previous chapter.

$$\frac{d}{dt}\begin{bmatrix} R_{13} \\ R_{23} \end{bmatrix} = \gamma \left(\begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \end{bmatrix} - \begin{bmatrix} R_{13} \\ R_{23} \end{bmatrix} \right). \tag{4.12}$$

Here γ^{-1} is an approximate time constant of the closed-loop attitude dynamics.

The lateral dynamics captured by equation (2.5) is, in fact, obtained from the assumption that the thrust vector lies perfectly along the body $\hat{z}$ axis. In other words, it assumes that the thrust vector in the inertial frame is given as

$$\begin{aligned} \frac{\Gamma}{|\Gamma|} &= \hat{z} \\ &= R \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T. \end{aligned} \tag{4.13}$$

To compensate for a possible misalignment, we introduce another rotation matrix R_ϵ representing a small rotation that maps the body $\hat{z}$ to the true direction of the thrust vector:

$$\frac{\Gamma}{|\Gamma|} = R R_\epsilon \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T. \tag{4.14}$$

For small rotations, R_ϵ can be parametrized by three parameters $|\epsilon_x|, |\epsilon_y|, |\epsilon_z| \ll 1$,

such that

$$R_\epsilon = \begin{bmatrix} 1 & -\epsilon_z & \epsilon_y \\ \epsilon_z & 1 & -\epsilon_x \\ -\epsilon_y & \epsilon_x & 1 \end{bmatrix} \tag{4.15}$$

Therefore, the complete model of the lateral dynamics is obtained by substituting equations (4.12), (4.14) and (4.15) into (2.5).

$$\begin{aligned} \gamma^{-1} m \frac{d^3}{dt^3} \begin{bmatrix} X \\ Y \end{bmatrix} + m \frac{d^2}{dt^2} \begin{bmatrix} X \\ Y \end{bmatrix} = \\ mg \begin{bmatrix} \hat{z}_{d1} \\ \hat{z}_{d2} \end{bmatrix} + \epsilon_x \begin{bmatrix} -R_{12} \\ -R_{22} \end{bmatrix} + \epsilon_y \begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}, \end{aligned} \tag{4.16}$$

where we have assumed R_ϵ to be constant, hence the two $\dot{\epsilon}$ terms arising from the differentiation of equation (4.14) are neglected.

Subsequently, the controller can be designed based on a similar composite variable idea as used for the attitude controller. The sliding surface $\mathbf{s}_l$, and the Lyapunov function candidate V_l are defined as

$$\begin{aligned} \mathbf{s}_l &= \left(\frac{d^2}{dt^2} + 2\lambda \frac{d}{dt} + \lambda^2 \right) \begin{bmatrix} \tilde{X} \\ \tilde{Y} \end{bmatrix} \\ V_l &= \frac{1}{2} \gamma^{-1} \mathbf{s}_l^T \mathbf{s}_l + \tilde{\beta}^T \Psi^{-1} \tilde{\beta}, \end{aligned} \tag{4.17}$$

where $\tilde{X}$ and $\tilde{Y}$ are position errors (the difference between the current position and the position setpoint), $\tilde{\beta}$ is a vector containing the estimation errors of γ^{-1}, ϵ_x, and ϵ_y, and Ψ is an adaptive gain. Given a positive diagonal controller gain matrix K_l, it

can be proven that the following control law

$$g \begin{bmatrix} \hat{z}_{d1} \\ \\ \hat{z}_{d2} \end{bmatrix} = -K_l \mathbf{s}_l + \frac{d^2}{dt^2} \begin{bmatrix} X \\ \\ Y \end{bmatrix} + \Upsilon \hat{\beta}, \tag{4.18}$$

with the term $\Upsilon\hat{\beta}$ written as

$$\Upsilon\hat{\beta} = \begin{bmatrix} \dfrac{d^3}{dt^3} \begin{bmatrix} X \\ \\ Y \end{bmatrix} - \dfrac{d}{dt}\mathbf{s}_l \end{bmatrix} \begin{matrix} R_{12}/m & -R_{21}/m \\ R_{22}/m & -R_{21}/m \end{matrix} \begin{bmatrix} \hat{\gamma}^{-1} \\ \\ \hat{\epsilon}_x \\ \\ \hat{\epsilon}_y \end{bmatrix}, \tag{4.19}$$

and the adaptive law

$$\dot{\hat{\beta}} = -\Psi \Upsilon^T \mathbf{s}_l, \tag{4.20}$$

make the time derivative of the proposed Lyapunov function candidate negative definite

$$\dot{V}_l = -\mathbf{s}_l^T K_l \mathbf{s}_l \leq 0. \tag{4.21}$$

Again, the invariant set theorem can be applied to ensure the stability of the system. In the case of hovering, the position setpoint is constant. In more general cases, the controller also possesses the ability to track time-varying setpoints as the first and second derivative of the setpoint are incorporated into the composite variable $\mathbf{s}_l$.

4.2.3 Adaptive Altitude Controller

The altitude controller has a structure similar to the lateral controller in the preceding section, but with only one dimensional dynamics and a feedforward term to

account for gravity. The input to the altitude dynamics is, however, the commanded thrust. The adaptive part is responsible for estimating the thrust offset and a time constant similar to γ in the lateral controller.

The main assumption on the altitude controller is that the robot orientation is always upright, thus the generated thrust is always aligned with the vertical axis. The primary reason for this assumption is to preserve the limited control authority for the more critical attitude controller. To illustrate, a tilted robot may lose altitude due to a reduction in thrust along the inertial vertical axis. Instead of producing more thrust to compensate, we prioritize control authority to the attitude controller to bring the robot upright and reorient the thrust to the vertical axis.

4.3 Unconstrained flight experiments

The lack of direct velocity and angular velocity measurement requires us to estimate both velocities via the use of filtered derivatives. The approach allows some attenuation of high frequency disturbances, but the estimates suffer from delays introduced by filter phase shifts. To illustrate, when a derivative filter $sa/(s+a)$ is applied, it introduces the attenuation factor of approximately a/ω to a high frequency (ω) signal, whrereas a low frequency signal would suffer a phase delay of approximately ω/a radians. This delay, however, was found to be sufficiently small and did not prevent us from achieving stable hover in the experiments.

As stated in the previous chapter, we belive the wire tether has a destabilizing effect on the robot, but it is compensated for by the controller. More thorough analysis on the tether is given in Appendix B.

4.3.1 Open-loop trimming

Initially, the vehicle is mounted on a static setup and a high-speed video camera is used to measure the flapping amplitude of the robot at various frequencies to determine the resonant frequency of the system and to characterize a suitable operating point where asymmetry between the two wings is minimized. Once the operating frequency is chosen, trimming flights are executed in the flight arena in attempt to determine the configuration where the net torque produced by the robot is close to zero.

Open-loop trimming is carried out by commanding the robot to produce constant thrust and torques. Visual feedback and state feedback are used to determine the amount of undesired bias torques present in the robot. The process is repeated with a new set of offset torques in an attempt to minimize the observed bias torques. Due to the inherent instability of the robot, a successful open-loop flight usually crashes in less than 0.4 s. This emphasizes the need of active control for the vehicle. An automatic switch-off routine is also implemented to cut off the power when the robot deviates more than 60° from vertical to prevent damages from crashing. An example of a well-trimmed open-loop flight, where the robot ascended to more than 4cm in altitude before crashing, is displayed in figure 4.2(a).

4.3.2 Attitude-Controlled flight

To evaluate the performance of the attitude controller, we first carried out closed-loop flights without the lateral controller by setting the attitude setpoint in equation (4.2) to be an upright orientation, bypassing the lateral controller block in figure 4.1.

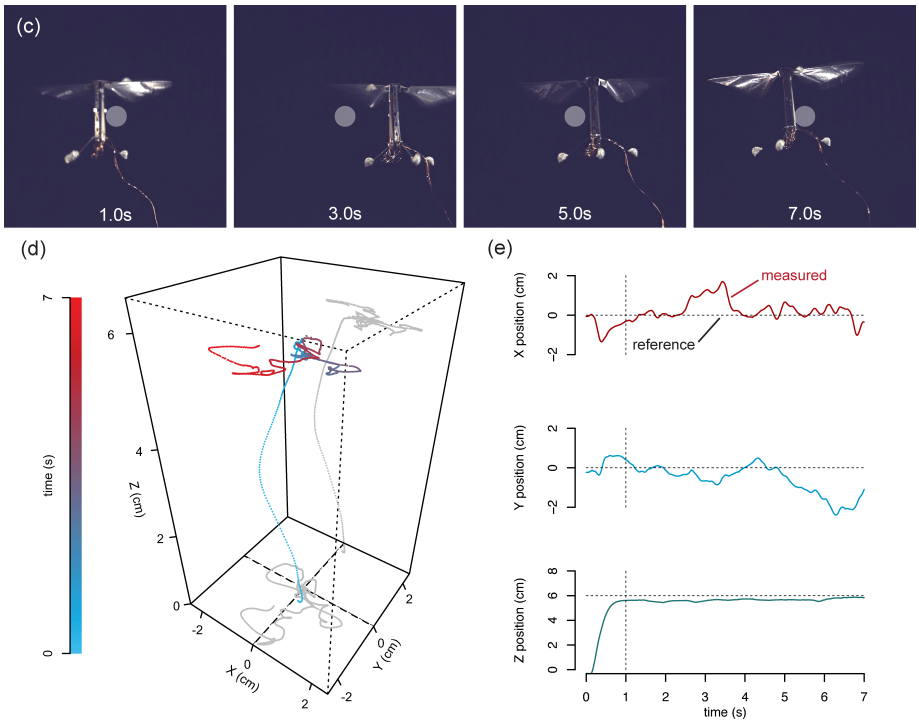

Figure 4.2: (a) Composite image showing an example of a trimming flight. (b) A composite image of a robotic insect taking of in a attitude-controlled flight. (c) Image sequence illustrating the robot hovering around the setpoint at various times (the grey dots indicate the approximate desired location). (d) A 3D reconstruction of the robot's trajectory during a seven-second hovering flight. (e) Position vs. time plots of the robot location in the same flight as in (c) and (d).

Theoretically, once the attitude and altitude are controlled, the robot should be able to stay aloft for an indefinite period.

Here, the trim information obtained from open-loop flights serves as an initial estimate of the torque biases for closed-loop flights. It was found that the attitude controller was able to keep the robot in the upright orientation as captured in figure 4.2(b). However, in the absence of the lateral controller, the robot generally traverses over 20 cm in $1 - 2$ s and immediately crashes once it is outside the tracking volume of the motion capture system. The results emphasize the need for a lateral controller to keep the robot flying in the control volume for an extended period.

4.3.3 Hovering flight

To begin with, hovering flights are performed using initial estimates of torque biases obtained from open-loop flights. At the beginning of each flight, only the attitude controller and the altitude controller are active. The robot usually takes less than one second to reach the altitude setpoint. The lateral controller is initiated 0.2 s into the flight, however, it is not fully activated until $t = 0.4$ s (The gain is ramped up in 0.2 s). Similarly, the adaptive parts are activated 0.8 s into the flight, but are not in full operation until 1.0 s.

Oftentimes, the parameter estimates derived from open-loop trimming flights are sufficiently accurate for the robot to stay aloft for a few seconds while the adaptive parts enhance the performance of the controller by adjusting these estimates. Nevertheless, parameter estimates learned at the end of each flight are incorporated into the controller as new estimates. These include the torque offset (τ_o) in the attitude

controller, orientation misalignment in the lateral controller (ϵ_x, ϵ_y), and the thrust offset in the altitude controller.

In the absence of mechanical fatigue, after several tuning flights, estimated parameters tend to converge to constant values. At this point, the robot is typically able to maintain its altitude setpoint within a few millimeters, while the lateral precision is on the order of one to two centimeters. It is likely that local air currents or tension from the power wires are the cause of disturbances.

Figure 4.2(c)-(e) demonstrates an example of a typical hovering flight after parameter convergence. In this case the flight lasts seven seconds, after which the power is cut off. In this sequence, the robot maintained an altitude of 6.0 cm above the ground while it translated laterally around the setpoint.

4.3.4 Lateral maneuver

To demonstrate tracking ability for both the attitude controller and the lateral controller and to verify the efficiency of accomplishing lateral maneuvers by tilting the body, we illustrate that simple lateral maneuvers along a pre-generated trajectory can be achieved. For this demonstration, a sinusoidal trajectory with the amplitude of 6.0 cm and the period of 3.0 s is chosen. The robot was configured to follow this trajectory for two cycles while retaining its altitude.

Figure 4.3(a)-(b) shows an example of a lateral flight maneuver. It reveals that throughout 8.0 s of the flight, the robotic insect could maintain its altitude within a few millimeters from the setpoint. The lateral position, however, had a tendency to lag behind the reference trajectory by approximately 0.2 s and occasionally overshot

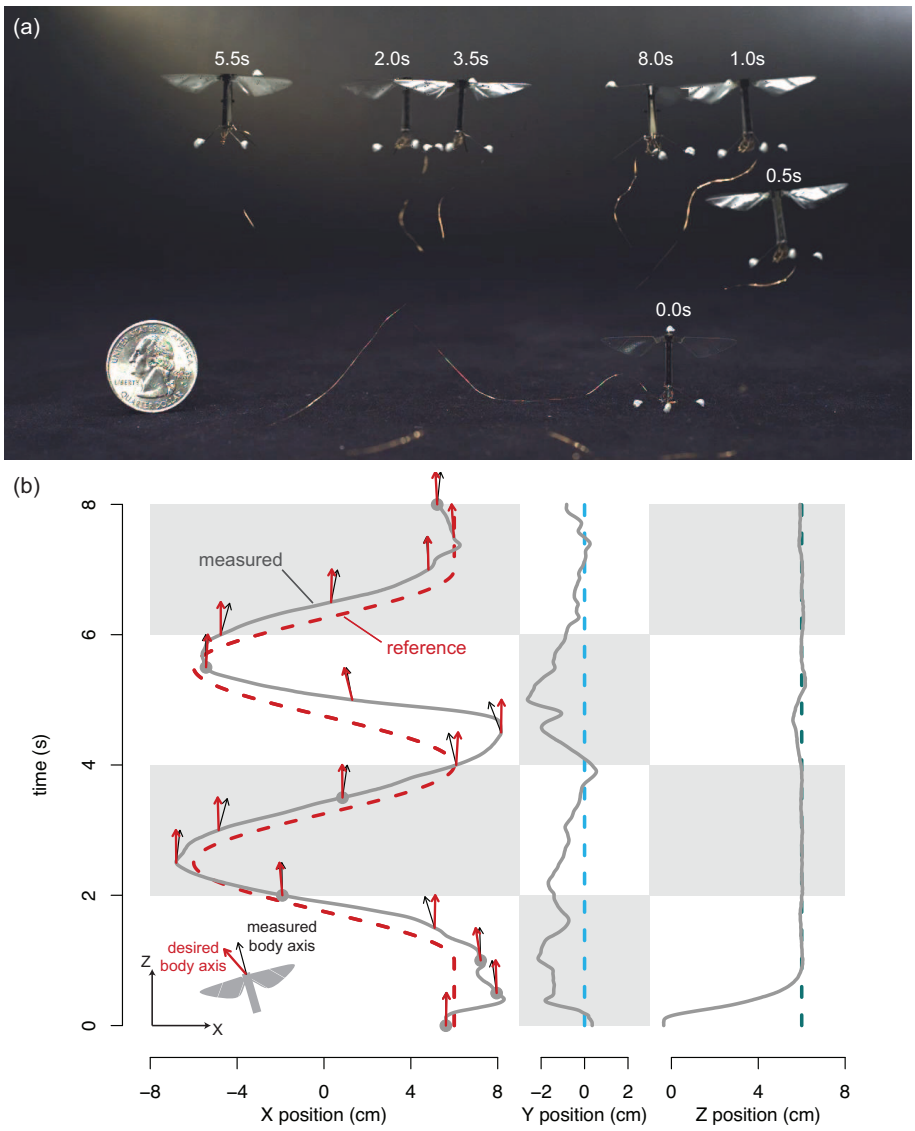

Figure 4.3: (a) A composite image showing the robot in various locations and times while performing a lateral maneuver. (b) A plot of the robot's position and desired location in a lateral flight. The orientations of the robot's body axis are projected onto the $X - Z$ plane.

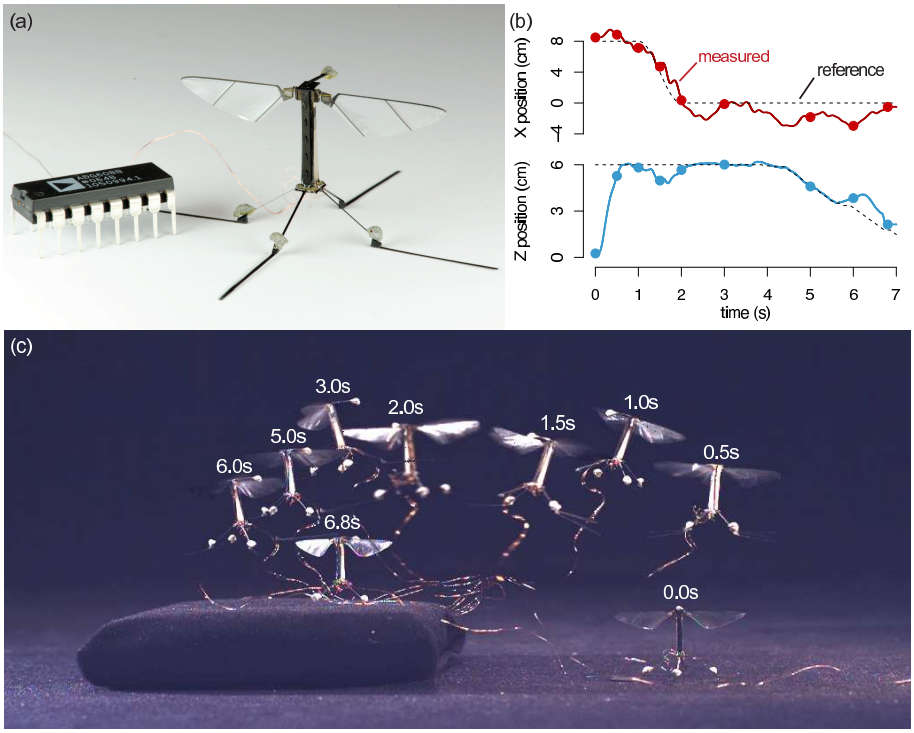

Figure 4.4: (a) Photograph of the robotic insect with extended landing gears next to a 16-pin dual in-line package (DIP) integrated circuit for scale. (b) Trajectory plot for an example landing flight. (c) An image illustrating the robot traversing laterally before landing on an elevated platform.

the target. This is in accord with the plot of the robot's orientations in figure 4.3(b), in which the robot often failed to catch up with the desired orientation when turning.

4.3.5 Vertical takeoff and landing

In order to avoid violent crashes and simultaneously demonstrate precise maneuvers, here we show a controlled takeoff and landing flight of a robotic insect. At the time of landing, the translational and angular velocities must be relatively small, otherwise the momentum would cause the robot to crash. Moreover, when the robot approaches the ground, downwash from the flapping wings may introduce distur-

bances in the form of ground effects as seen in larger flying vehicles, and destabilize the robot. Here, we illustrate successful landing flights via the use of a simple control strategy with the aid of mechanical landing gear.

The landing gear is designed with two goals: to widen the base of the robot and to absorb the impact of landing. Carbon fiber extensions are attached to the existing structure through a viscoelastic urethane spacer (Sorbothane). A photograph of a robot with the additional landing gear is shown in figure 4.4(a).

Landing is achieved by slowly reducing the altitude setpoint. To ensure that the robot remains in the nominal upright orientation and stays close to the lateral setpoint, the change in altitude setpoint is suspended when the vehicle is in an unstable state, defined as the l_2-norm of the composite variable $\mathbf{s}_a$ or $\mathbf{s}_l$ being larger than the chosen thresholds. Once the robot is less than a certain height (≈ 8 mm) above the ground, the driving signals are ramped down, leaving the landing gear to absorb the impact from falling.

An example trajectory of a successful vertical takeoff and landing flight of the robotic insect is displayed in figure 4.4(b)-(c). In this case, the robot took off towards the altitude setpoint at 6 cm and started the landing process just before 1.0 s. The nominal landing speed was set at 1.5 cm$\cdot$s^{-1}. According to the plot, the robot followed the altitude setpoint closely. Nevertheless, just after $t = 5.5$ s, it can be seen that the landing was briefly suspended as the vehicle drifted away from the lateral setpoint beyond the set tolerance. Eventually, the robot reached the pre-defined landing altitude and the power was ramped down after six seconds.

4.4 Conclusion and Discussion

We have presented a comprehensive flight controller designed for a bio-inspired flapping-wing microrobot. Driven by modeling uncertainty and the nonlinear nature of the system, Lyapunov's direct method was employed to guarantee the stability of the proposed adaptive controllers. We have demonstrated that, using a simplified set of wing kinematics, as opposed to more sophisticated wing kinematics observed in real insects, the proposed controller is sufficient to realize hovering flights and simple maneuvers.

4.4.1 Comparison to the non-adaptive controller

To compare the proposed controller to its non-adaptive counterpart from Chapter 3, we consider 25 flight trajectories from the non-adaptive controller and 28 flight trajectories obtained from the adaptive controller. Only portions of stable flights after taking off are selected and consolidated into a non-adaptive dataset and an adaptive dataset. The non-adaptive dataset consists of 125 seconds (15,000 wingbeats) of flying time while the adaptive dataset is 115 seconds (13,800 wingbeats) long. These trajectories are plotted along the X, Y, and Z axes relative to the setpoints in figure 4.5(a). They are overlaid by boxplots representing the mean positions and the standard deviations. It can be seen that the proposed adaptive controller markedly reduces standard errors by over 50%. The improvement is most pronounced along the $\hat{Z}$ direction, thanks to the adaptive altitude controller.

Furthermore, to validate that the proposed adaptive controller is capable of coping with possible variations between robots, we carried out hovering flight experiments on

two additional robots of the same design. On the second robot, data from nine stable flights, totaling 44.5 seconds (5,340 wingbeats) of flight time was captured. On the third robot, seven stable flights were obtained, with total flight time of 45.5 seconds (5,460 wingbeats). Figure 4.5(b) shows boxplots illustrating the mean positions and the standard deviations of stable hovering flights performed by the three robots using the adaptive controller. The plots verify that the magnitudes of the means and the standard deviations are similar and consistent between the robots. Additionally, we also found that in these robots, the torque offsets converged to markedly different values. For instance, the pitch torque offsets of the second and the third robot were found to be ≈ -0.45 units and ≈ -1.50 units respectively, while the pitch torque offset of the first robot (as demonstrated in figure 4.5c) was ≈ -1.00 units. Note that the torque values here are given in a normalized unit as the true values are not measured. Nevertheless, one unit is estimated to be in the order of one μNm. The results suggest that the adaptive component of the controller is able to overcome variations due to fabrication imperfections between robots.

Computationally, the adaptive laws of the proposed controller have a simple structure comparable to the integral component of a PID controller. The parameter estimates are given by the integral of terms without any matrix inversion or computationally expensive operations (equations 4.6 and 4.20), resulting in no significant computing power requirement relative to a standard PID controller.

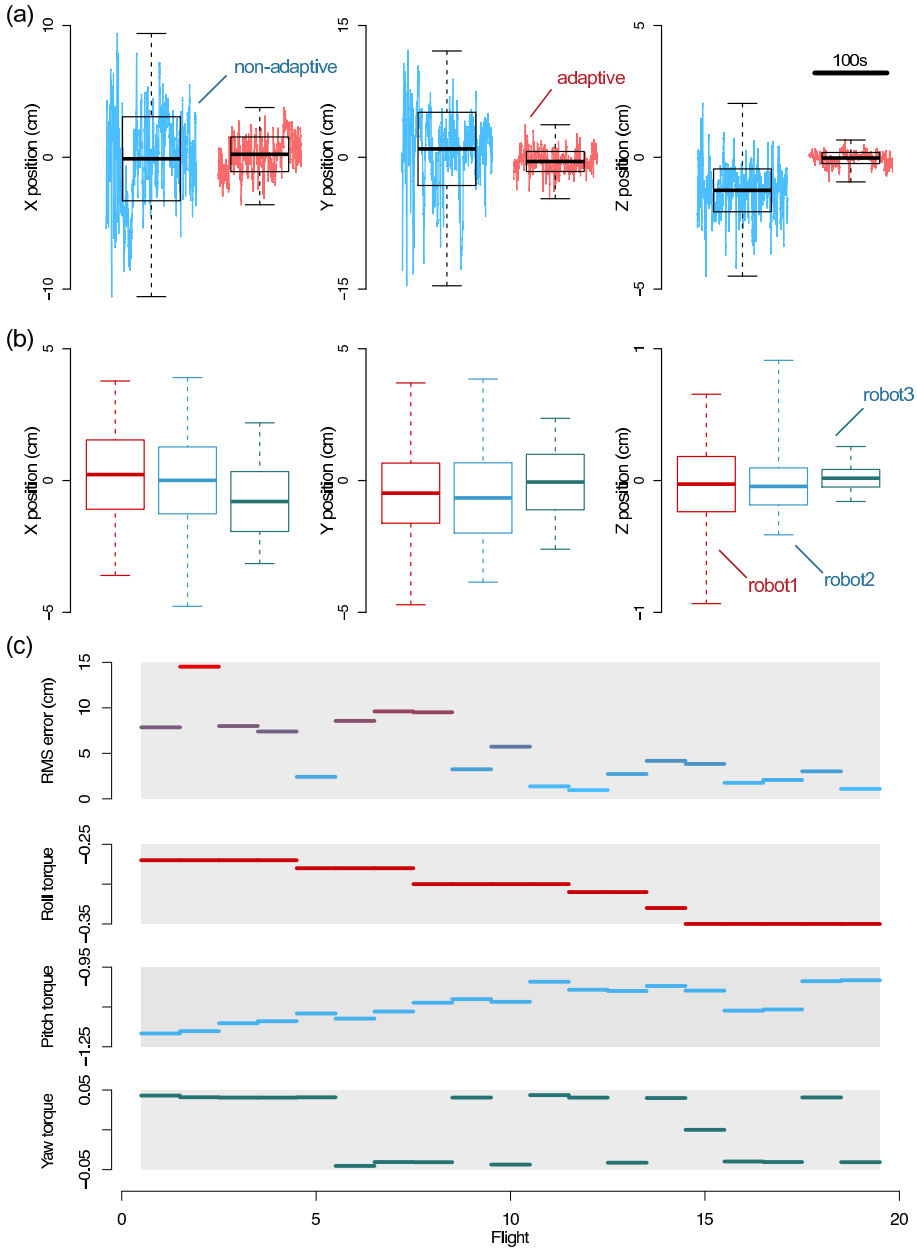

Figure 4.5: (a) Positions of the robot with respect to the setpoints from 25 non-adaptive flights (blue) and 28 adaptive flights (red) overlaid by boxplots showing the mean positions and the standard deviations. (b) Boxplots showing the averages and the standard deviations in positions of three robots from 28, 9 and 7 flights. (c) Data from 19 consecutive flights after the open-loop trimming for steady hovering. The top plot shows the RMS errors of the lateral position. The bottom three plots show the estimated torque offsets at the beginning of each flight.

4.4.2 Adaptability

Previous work on control for flapping-wing MAVs has typically been carried out through modeling and simulations [67]. As a result, less emphasis was given to uncertainties in state estimation. Some previous research incorporated integral parts into linear controllers to eliminate steady state errors [19, 34]. It is anticipated that this strategy may work equally well near hovering conditions. Due to the inherent nonlinearity of the attitude dynamics, the proposed adaptive controller has the potential to surpass linear controllers at other operating points, when the linearization becomes less accurate. It follows that the adaptive controller concept might resemble insect flight control systems more closely than a PID controller since flying insects often perform saccadic movements and highly aggressive maneuvers, experiencing conditions that are remarkably dissimilar to the hovering state.

Regarding the adaptive ability, one consideration is the extent of modeling uncertainties that the adaptive controller can compensate for. In practice, it was found that it was always necessary to perform open-loop trimming flights to obtain the approximate nominal torque offsets (τ_o). This serves as a starting point for a robot to fly short closed-loop flights prior to crashing. Subsequent flights were executed with the adaptive components and the latest parameter estimates were rounded and used as new starting points for later flights. Figure 4.5(c) shows the experimental data from 19 consecutive flights after open-loop trimming. By continuously updating the parameter estimates (in this case the torque offset estimates at the beginning of each flight are shown), the root mean square of the lateral position error decreases from more than 10cm to less than 2 cm. Similarly, the torque offset estimates seem to

eventually converge. The adaptive component is capable of correcting torque offset estimates that are up to 0.2 units, whereas the inherent offset could be as much as one unit. To put these numbers into perspective, for a stable hovering flight, the commanded torques from the controller are generally bounded within ± 0.3 units.

In practice, we occasionally observed unanticipated (both gradual and sudden) changes in the torque offset values caused by mechanical changes in the robot (e.g. wing damage or wing hinge fatigue). Due to the small scale, it is often not possible to identify the source of such changes immediately by inspection. To illustrate, a small crack in the wing hinge is usually not observable until it has propagated substantially. While the adaptive component of the controller could usually deal with small and gradual changes as the damage propagates, it is difficult to quantify the damage in terms of the amount of imbalanced torque it causes as we are unable to identify the beginning of the failure. In a hypothetical event of slight damage on the one wing, we consider a simulated scenario involving a reduction of lift from one wing. Typically one wing generates ≈ 40 mg of lift to support the weight of the robot. Assuming the center of pressure is approximately one centimeter from the center of mass, each wing produces about 4 μNm of roll torque. The results shown in figure 4.5(c) suggests that the adaptive controller would be able to cope with ≈ 0.20 μNm of imbalanced roll torque, which may arise from a sudden 5% reduction in lift from one wing. In a scenario where the damage is gradual as often observed in practice, we expect that the adaptive component would be able to deal with more acute damage.

In our adaptive controller, the adaptive gain can be chosen. In experiments, we have seen that the torque offset estimates could be adapted as fast as 2 unit$\cdot$s^{-1}

without losing stability. This is, in many circumstances, sufficiently fast for the robot to adapt for changes that occur during flight (e.g. changes in torque offsets, loss of lift, and the presence of gradual fatigue at the wing hinge). In insects, similar learning behavior has been observed [22, 11]. Insects were found to adapt their flight behaviors to cope with wing damage with noticeable effects on flight performance. The results in this work are in agreement with these biological observations, suggesting the presence of an inherent adaptive or learning ability in the insect flight control systems.

4.4.3 Robustness Analysis

Here we offer a simplified analysis to quantify the effect of unaccounted for torque offsets on the flight performance. This could also be seen as a theoretical approach to bound the position errors from possible unmodeled dynamics.

To begin with, we exclude the adaptive part of the controller from the analysis, or regard the adaptive gains as zero. For the attitude controller, equations (4.6) and (4.11) reveal that the time derivative of the Lyapunov function candidate is no longer always negative definite when $\tilde{\tau}_o > K_a \mathbf{s}_a$. Suppose the robot possesses an inherent torque offset of 0.1units about the pitch or roll axes, this would translate to a consistent attitude error of 6.5° for the controller gains used in the experiments. Treating that as a misalignment of body axes for the lateral controller, the resultant error in lateral position would be 3cm. Since this does not take into consideration other unmodeled effects (e.g. disturbances or a delay in the control loop), it is reasonable to anticipate the error to be larger in practice. This is approximately in accordance with the data in figure 4.5(b), where the lateral error drops from ≈ 10 cm to < 2 cm

when the torque offset estimates were altered by ≈ 0.2 units.

At the scale of our robot, to obtain millimeter-scale spatial accuracy while hovering on par with insects, the adaptive component must correct for torque offsets with a resolution of approximately 0.01 μNm resolution. This suggests that some insects are capable of recognizing extremely small steady state errors and finely tuning their wing kinematics in response. Since the adaptive process involves integration and does not require rapid feedback, it is likely that insects handle such operation in the primary sensory-motor systems, rather than using the specialized low-latency sensory-based equilibrium reflexes typically used for attitude stabilization [20, 83].

Chapter 5

Adaptive Tracking Control

Inspired by the agility of flying insects and the recent development on an insect-scale aerial vehicle, we propose a single-loop adaptive tracking flight control suite designed with an emphasis on the ability to track dynamic trajectories as a step towards the goal of performing acrobatic maneuvers as observed in real insects. Instead of the conventional approach of having cascaded control loops, proposed controller directly regulates the commanded torques to stabilize the attitude and lateral position in a single loop. This method is verified by performing trajectory following flights with the insect-like robot. The results show that the position errors during trajectory following flights are markedly reduced from those performed by the adaptive controller presented in Chapter 4.

5.1 Introduction

In an effort to improve the flight performance, in the previous chapter, the lack of comprehensive knowledge of the system and variation caused by imperfect fabrication motivated the development of an adaptive flight controller capable of coping with model uncertainties. This has brought about marked improvement in flight performance as evidenced by a reduction in position errors, particularly for hovering flights. Moreover, the fidelity of this flight controller has been further demonstrated in vertical takeoff and landing flights.

In flight control of MAVs with similar dynamics–quadrotors, it is common to divide the controller into an inner loop and an outer loop [60, 2, 43]. This also applies to the previous controllers presented in Chapters 3 and 4. This approach relies on the assumption that the dynamics of the inner loop are significantly faster than for the outer loop, hence two loops could be arranged in a cascaded configuration. In the case of Chapter 4, the inner loop, which control the attitude dynamics, takes the output from the lateral controller in the form of an attitude setpoint as its input. While this was shown to be an effective method to produce steady hovering flights, it is conceivable that the cascaded control architecture may lead to unavoidable delay due to the mentioned assumption, rendering such a controller unsuitable for more aggressive flight trajectories.

As a consequence, in this chapter we present the development of a flight controller that discards the cascaded structure, integrating the lateral controller and the attitude controller into a single block. Not only does this eliminate the delay in the control loop, but it also effectively improves the position tracking ability by taking into

account the third and fourth order derivatives of the desired position while generating the control outputs. We verified the capability of the proposed controller in trajectory following flights using generated smooth trajectories outlined in the later part of the chapter.

5.2 Adaptive Tracking Controller Design

The inherent instability of flapping-wing MAVs [75, 67] requires active flight control. To prevent the robot from crashing, the attitude controller must nominally align the robot's thrust vector against gravity. In Chapter 3, we have demonstrated that, using a flight controller that possessed a large region of attraction over the $\mathrm{SO}(3)$ space, the flapping-wing robot achieved stable hovering flights. One distinctive character of the proposed controller is the relaxation of control over the exact yaw orientation as it is dispensable in controlling the lateral position of the vehicle as described in equation (2.5).

To attain more precise hovering, we have identified several critical unknown parameters that significantly affect the flight performance and re-designed an adaptive flight controller using sliding mode control techniques as shown in Chapter 4. In this case, the lateral position of the robot is regulated by changing the attitude setpoint of the robot, while the altitude is controlled separately. In other words, the lateral controller and the attitude controller operate in a cascaded fashion. The lateral controller determines the attitude setpoint by assuming that the attitude dynamics are considerably faster than the lateral dynamics, and therefore the closed-loop attitude dynamics can be treated as a first order lowpass system. In the mean time, the at-

titude controller attempts to realize the attitude setpoint and minimize the angular velocity of the robot. The block diagram summarizing key components of this control architecture is shown in figure 5.1. This markedly improved the accuracy in position and substantially reduced visible oscillations during hovering flights as compared with the results obtained from the non-adaptive controller presented in Chapter 3. Simple lateral maneuvers were also demonstrated, nonetheless, there was significant room for improvement.

One downside of having a cascaded control structure as in Chapter 3 and 4 and other related literature [2, 60] is a possible loss in precision due to the assumption that the inner control loop is considerably faster than the outer loop. Another drawback of the proposed controller in chapter 4 is that the attitude controller always tries to minimize the rotational rate. Since the rotational rate is related to the third order derivative of the position, it is anticipated that a controller that also determines a suitable angular velocity setpoint would bring about better performance in trajectory following, particularly when more aggressive movements are involved.

In [53], a nonlinear controller that explicitly tracks trajectories in $\mathrm{SE}(3)$ was proposed and proved to have exponential attractiveness over a large region. This controller requires a pre-defined trajectory that includes reference position, orientation, and angular velocity information. A variation of this controller was implemented for quadrotors in [59]. In our case, previously developed controllers possess two primary shortcomings. First, it is not trivial to evaluate the angular velocity setpoint from the pre-planed trajectory and the current state of the vehicle when the heading is not directly specified. Second, they lack adaptibility, which is a desired property that

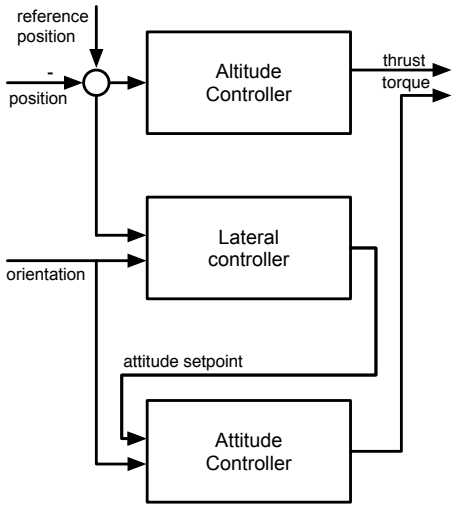

Figure 5.1: A block diagram illustrating the structure of the flight controller in Chapter 4. Here the lateral controller computes the attitude setpoint as an input for the attitude controller.

was proved to be crucial for flight performance at this scale. It is not trivial to design a controller with proven stability that satisfies the mentioned specifications. The complication arises as we try to retain provable Lyapunov stability while the exact yaw orientation of the robot is not directly controlled. In this section, we propose a Lyapunov function composed of variables made of various derivatives of position error projected on to suitable directions. The outcome is a tracking controller that directly regulates the desired torques from the position error, eliminating the former cascaded structure and, thus, dropping the assumption regarding the response of the attitude controller. The product is a more versatile controller that is capable of more aggressive trajectory following in addition to only steady hovering without sacrifying adaptibility.

In this section, we present the derivation of the proposed altitude controller and the trajectory tracking controller based on techniques borrowed from the sliding mode

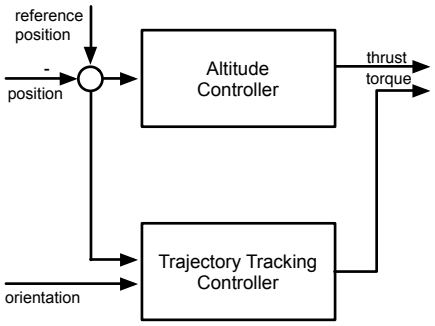

Figure 5.2: A simplified block diagram showing the underlying structure of the proposed single-loop controller. The attitude controller is incorporated into the lateral controller which operates in parallel to the altitude controller.

control method. Although they are presented separately as illustrated in figure 5.2, they operate in parallel rather than in a cascaded configuration. As a consequence, they can technically be classified as a single control loop. For clarity, we initially restrict the presentation to a non-adaptive system. Later, the controller is extended to accommodate an adaptive component and the stability is verified via Lyapunov's direct method.

5.2.1 Altitude Control

To begin, we define a position vector ($\mathbf{r}$) and the desired position vector ($\mathbf{r}_d$) with respect to the inertial frame:

$$\begin{aligned} \mathbf{r} &= \begin{bmatrix} X & Y & Z \end{bmatrix}^T \\ \mathbf{r}_d &= \begin{bmatrix} X_d & Y_d & Z_d \end{bmatrix}^T . \end{aligned} \tag{5.1}$$

Given the robot's normalized thrust Γ and the gravity vector $\mathbf{g}$ from equation (2.5), the translational dynamics of the robot , ignoring the extra aedodynamic term, are

described by

$$\begin{aligned} \ddot{\mathbf{r}} &= \Gamma\hat{z} + \mathbf{g} = \Gamma R \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T + \mathbf{g} \\ &= \Gamma \begin{bmatrix} R_{13} & R_{23} & R_{33} \end{bmatrix}^T - \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T . \end{aligned} \tag{5.2}$$

The altitude dynamics are given by the third row of equation (5.2).

$$\ddot{Z} = \ddot{\mathbf{r}} \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T = \Gamma R_{33} - g \tag{5.3}$$

The thrust produced by the robot Γ is modeled to be related to the commanded thrust T by a first-order differential equation and a gain factor γ to capture the effect of the actuation delay similar to the consideration in equation (4.12):

$$\dot{\Gamma} = \gamma \left(T - \Gamma \right). \tag{5.4}$$

For control purposes, we define an auxiliary variable S_Γ and the variable $\ddot{Z}_r$ as the following:

$$\begin{aligned} S_\Gamma &= \left(\ddot{Z} - \ddot{Z}_d \right) + \Lambda_1 \left(\dot{Z} - \dot{Z}_d \right) + \Lambda_2 \left(Z - Z_d \right) \\ &= \ddot{Z} - \ddot{Z}_r, \end{aligned} \tag{5.5}$$

where Λ_i's are positive constants. The sliding surface is described by $S_\Gamma = 0$. According to equations (5.3)-(5.5), the time derivative of S_Γ is

$$\begin{aligned} \dot{S}_\Gamma &= \dot{\Gamma} R_{33} + \Gamma \dot{R}_{33} - \frac{d}{dt}\ddot{Z}_r \\ &= \gamma \left(T - \Gamma\right) R_{33} + \Gamma \left(-R_{32}\omega_x + R_{31}\omega_y\right) - \frac{d}{dt}\ddot{Z}_r. \end{aligned} \tag{5.6}$$

Here we propose a Lyapunov function candidate

$$V_\Gamma = \frac{1}{2} S_\Gamma^2. \tag{5.7}$$

Subsequently, the following control law

$$T = \Gamma - \gamma^{-1} R_{33}^{-1} \left[\Gamma \left(-R_{32}\omega_x + R_{31}\omega_y\right) - \frac{d}{dt}\ddot{Z}_r + K_\Gamma S_\Gamma\right], \tag{5.8}$$

with a positive constant gain K_Γ and the measured thrust from equation (5.4) given as $\Gamma = \|\ddot{\mathbf{r}} + \mathbf{g}\|$ render the derivative of the Lyapunov function candidate negative definite

$$\dot{V}_\Gamma = S_\Gamma \dot{S}_\Gamma = -K_\Gamma S_\Gamma^2 \leq 0 \,. \tag{5.9}$$

According to the invariant set theorem, the system is globally asymptotically stable in a Lyapunov sense [80].

5.2.2 Trajectory Tracking Control

Since the angular velocity is related to the third-order derivative of the robot's position, we consider an auxiliary variable $\mathbf{e}$, made up of the differences between the

robot's position and the setpoint and their derivatives.

$$\begin{aligned} \mathbf{e} &= \left(\mathbf{r}^{(3)} - \mathbf{r}_d^{(3)}\right) + \lambda_1 \left(\ddot{\mathbf{r}} - \ddot{\mathbf{r}}_d\right) + \lambda_2 \left(\dot{\mathbf{r}} - \dot{\mathbf{r}}_d\right) + \lambda_3 \left(\mathbf{r} - \mathbf{r}_d\right) \\ &= \mathbf{r}^{(3)} - \mathbf{r}_r^{(3)}, \end{aligned} \tag{5.10}$$

where we have defined $\mathbf{r}_r^{(3)}$ accordingly. Using equation (2.4) and (5.2), the third derivative of $\mathbf{r}$ then becomes

$$\mathbf{r}^{(3)} = \Gamma \dot{\hat{z}} + \dot{\Gamma} \hat{z} = \Gamma \left(-\omega_x \hat{y} + \omega_y \hat{x}\right) + \dot{\Gamma} \hat{z}. \tag{5.11}$$

We propose the following composite variable $\mathbf{S}_\tau$ and the Lyapunov function candidate V_τ:

$$\begin{aligned} \mathbf{S}_\tau &= \begin{bmatrix} -\mathbf{e} \cdot \hat{y}/\Gamma & \mathbf{e} \cdot \hat{x}/\Gamma & \omega_z \end{bmatrix}^T \\ &= \begin{bmatrix} \omega_x + \Gamma^{-1} \left(\mathbf{r}_r^{(3)} \cdot \hat{y}\right) \\ \omega_y - \Gamma^{-1} \left(\mathbf{r}_r^{(3)} \cdot \hat{x}\right) \\ \omega_z \end{bmatrix}, \end{aligned} \tag{5.12}$$

$$V_\tau = \frac{1}{2} \mathbf{S}_\tau^T J \mathbf{S}_\tau. \tag{5.13}$$

Notice that angular velocity terms appear in (5.12), linking the attitude dynamics to the lateral dynamics. The third element of $\mathbf{S}_\tau$ is chosen to be ω_z, consistent with the decision not to directly control the heading of the robot, but only to damp out the yaw rotation rate that may arise. Using equations (2.2) and (5.12), we can write the derivative of the composite variable as

$$J\dot{\mathbf{S}}_\tau = \tau - (\omega \times J\omega) - \Gamma^{-1} J \frac{d}{dt} \begin{bmatrix} -\left(\mathbf{r}_r^{(3)} \cdot \hat{y}\right) \\ \left(\mathbf{r}_r^{(3)} \cdot \hat{x}\right) \\ 0 \end{bmatrix}.$$

This suggests the commanded body torque

$$\tau = -\Gamma^{-1} \begin{bmatrix} \mathbf{r}_r^{(3)} \cdot \hat{y} \\ -\mathbf{r}_r^{(3)} \cdot \hat{x} \\ 0 \end{bmatrix} \times J\omega + \Gamma^{-1} J \frac{d}{dt} \begin{bmatrix} -\left(\mathbf{r}_r^{(3)} \cdot \hat{y}\right) \\ \left(\mathbf{r}_r^{(3)} \cdot \hat{x}\right) \\ 0 \end{bmatrix} - K_\tau \mathbf{S}_\tau, \quad (5.14)$$

so that the time derivative of the proposed Lyapunov function candidate is negative definite and the system is proven asymptotically stable:

$$\begin{aligned} \dot{V}_\tau &= -\mathbf{S}_\tau^T K_\tau \mathbf{S}_\tau - \mathbf{S}_\tau^T (\mathbf{S}_\tau \times J\omega) \\ &= -\mathbf{S}_\tau^T K_\tau \mathbf{S}_\tau \leq 0. \end{aligned} \quad (5.15)$$

To evaluate the domain of attraction of the controller, observe that the Lyapunov function candidate in equation (5.13) is zero only when the robot has no angular velocity ω_z, and the auxiliary variable $\mathbf{e}$ is zero or parallel to the $\hat{z}$-axis. It is zero only when the robot tracks the reference trajectory perfectly, or $\mathbf{e}$ being parallel to the $\hat{z}$ axis implies that the error is along the thrust direction, in which case it will be taken care of by the altitude controller as demonstrated in Section 5.2.1. The exception occurs when the vector $\mathbf{e}$ points in the opposite direction to the $\hat{z}$-axis. In that circumstance, the control signal makes no attempt to generate any torques and correct for the error. Therefore, the proposed control law is almost globally

asymptotically stable. One contribution to the large region of attraction achieved here is owing to the absence of singularities associated with representations of SO (3) such as Euler angles or ambiguities of quaternions.

Examining the control law in equation (5.14), it can be seen that the third derivative of the measurement of $\mathbf{r}$ embedded in the term $-K_\tau \mathbf{S}_\tau$ could be written in terms of angular velocity as given in equation (5.11). Therefore, only the second derivative of $\mathbf{r}$ is required, alongside the body orientation and its first derivative. Furthermore, that fact that $\mathbf{r}_d^{(3)}$ and $\ddot{\mathbf{r}}_d$ is included in $\mathbf{r}_r^{(3)}$ implies that the controller effectively tracks a setpoint for the angular velocity in addition to the acceleration—the property lacking in the previous controller shown in Chapter 4.

5.2.3 Adaptive Control

In Chapter 4, we identified six unknown parameters that were crucial to accomplish steady hover: the misalignment of the thrust vector from the $\hat{k}$ axis (ϵ_i and ϵ_j), three unknown torque offsets ($\tau_o = \begin{bmatrix} \tau_{oi} & \tau_{oj} & \tau_{ok} \end{bmatrix}^T$), and the normalized thrust offset (T_o). In this section, we present how the proposed controller is modified to take into account the effects of these unknowns, starting by including those effects into the dynamic model. We then present how the composite variables and the control laws should be altered. A predictor and an adaptive component are implemented to ensure that the estimates of the unknowns converge to their true values and stability is still guaranteed under some assumptions as shown in the Lyapunov analysis at the end of the section.

5.2.3.1 Altitude control law

For a small deviation of the thrust vector from the presumed robot $\hat{z}$-axis, the thrust takes on small lateral components along the $\hat{x}$ and $\hat{y}$ axes, resulting in a slight modification to equation (5.2),

$$\ddot{\mathbf{r}} = \mathrm{r}\left(\hat{z} + \epsilon_x \hat{x} - \epsilon_y \hat{y}\right) - \mathbf{g}. \tag{5.16}$$

Similarly, the thrust dynamics are modified to include the unknown offset T_o by substituting T by $T_c - T_o$, $\dot{T} = \gamma\left(T_c - T_o - \Gamma\right)$, where T_c is the commanded thrust input. The derivative of the composite variable defined in equation (5.5) becomes

$$\begin{aligned} \dot{S}_\Gamma &= \gamma\left(T_c - T_o - \Gamma\right)\left(R_{33} + \epsilon_x R_{31} - \epsilon_y R_{32}\right) + \Gamma\left(-R_{32}\omega_x + R_{31}\omega_y\right) \\ &\quad + \Gamma\epsilon_x\left(-R_{33}\omega_y + R_{32}\omega_z\right) + \Gamma\epsilon_y\left(-R_{33}\omega_x + R_{31}\omega_z\right) - \frac{d}{dt}\ddot{Z}_r. \end{aligned} \tag{5.17}$$

We define

$$\begin{aligned} \mu &= \Gamma\hat{\epsilon}_x\left(-R_{33}\omega_y + R_{32}\omega_z\right) + \Gamma\hat{\epsilon}_y\left(-R_{33}\omega_x + R_{31}\omega_z\right) \\ &\quad + \Gamma R_{31}\omega_y - \Gamma R_{32}\omega_x - d\ddot{Z}_r/dt, \end{aligned} \tag{5.18}$$

and propose the following control signal

$$T_c = \hat{T}_o + \Gamma - \Gamma^{-1}R_{33}^{-1}\left(1 - \hat{\epsilon}_x R_{31}R_{33}^{-1} + \hat{\epsilon}_y R_{32}R_{33}^{-1}\right)\left(\mu + K_\Gamma S_\Gamma\right). \tag{5.19}$$

By using the approximation

$$\left(1 - \hat{\epsilon}_x R_{31} R_{33}^{-1} + \hat{\epsilon}_y R_{32} R_{33}^{-1}\right) = \left(1 + \hat{\epsilon}_x R_{31} R_{33}^{-1} + -\hat{\epsilon}_y R_{32} R_{33}^{-1}\right)^{-1} + \mathcal{O}\left(\hat{\epsilon}^2\right) \qquad (5.20)$$

it can be shown that the proposed control law makes the derivative of the composite variable in equation (5.17) expressible as

$$\begin{aligned}
\dot{S}_\Gamma &= -K_\Gamma S_\Gamma + \begin{bmatrix} \gamma\left(R_{33} + R_{31}\hat{\epsilon}_x - R_{32}\hat{\epsilon}_y\right) \\ \Gamma R_{33}\omega_y - \Gamma R_{32}\omega_z - R_{31}R_{33}^{-1}\mu \\ \Gamma R_{33}\omega_x - \Gamma R_{31}\omega_z - R_{32}R_{33}^{-1}\mu \end{bmatrix}^T \begin{bmatrix} \tilde{T}_o \\ \tilde{\epsilon}_x \\ \tilde{\epsilon}_y \end{bmatrix} + \gamma\tilde{T}_o\left(-R_{31}\tilde{\epsilon}_x + R_{32}\tilde{\epsilon}_y\right) \\
&= -K_\Gamma S_\Gamma + Y_\Gamma \tilde{\mathbf{a}} + \gamma\tilde{T}_o\left(-R_{31}\tilde{\epsilon}_x + R_{32}\tilde{\epsilon}_y\right), \qquad (5.21)
\end{aligned}$$

where we have defined $\mathbf{a}$ as a vector consisting of the three unknown parameters and the vector Y_Γ accordingly. For small deviations ($\epsilon_x, \epsilon_y \approx 0.1$), and a reasonably large tilt angle ($R_{31}R_{33}^{-1}, R_{32}R_{33}^{-1} \approx 1$) the error caused from the approximation in equation (5.20) is less than one percent. The first term in equation (5.21) is identical to the non-adaptive case in equation (5.9). The vector Y_Γ in the second term contains only known and measurable quantities. These two terms are the typical form that usually appears in the derivation of an adaptive sliding mode controller [80]. To achieve Lyapunov stability, the last term has to be handled explicitly described in the last paragraph of Section 5.2.3.4.

5.2.3.2 Trajectory tracking control law

Including the effect of ϵ_x and ϵ_y, we define an estimate of $\mathbf{r}^{(3)}$ based on the estimates of ϵ_x and ϵ_y based on equation (5.11):

$$\hat{\mathbf{r}}^{(3)} = \Gamma\left(\dot{\hat{z}} + \hat{\epsilon}_x\dot{\hat{x}} - \hat{\epsilon}_y\dot{\hat{y}}\right) + \dot{\Gamma}\left(\hat{z} + \hat{\epsilon}_x\hat{x} - \hat{\epsilon}_y\hat{y}\right). \tag{5.22}$$

It follows that we can also define an estimate of $\mathbf{e}$ from equation (5.10) as $\hat{\mathbf{e}} = \hat{\mathbf{r}}^{(3)} - \mathbf{r}_r^{(3)}$. The definition of $\mathbf{r}_r^{(3)}$ from equation (5.10) remains unchanged. As a consequence, the composite variable of the tracking controller is re-defined:

$$\begin{aligned}\hat{\mathbf{S}}_\tau &= \begin{bmatrix} -\hat{\mathbf{e}}\cdot\hat{y}/\Gamma & \hat{\mathbf{e}}\cdot\hat{x}/\Gamma & \omega_z \end{bmatrix} \\ &\quad + \begin{bmatrix} \omega_x - \hat{\epsilon}_x\omega_z + \Gamma^{-1}\left(\mathbf{r}_r^{(3)}\cdot\hat{y}\right) \\ \omega_y + \hat{\epsilon}_y\omega_z - \Gamma^{-1}\left(\mathbf{r}_r^{(3)}\cdot\hat{x}\right) \\ \omega_z \end{bmatrix} + \frac{\dot{\Gamma}}{\Gamma}\begin{bmatrix} \hat{\epsilon}_y \\ -\hat{\epsilon}_x \\ 0 \end{bmatrix}\end{aligned} \tag{5.23}$$

In general maneuvers, Γ does not vary appreciably from g. The last term in equation (5.23) can be neglected.

To avoid the presence of $\dot{\mathbf{r}}_r^{(3)}$—which includes unknown parameters—in the control law, we define $\dot{\hat{\mathbf{r}}}_r^{(3)}$ from equation (5.10) and (5.22) as

$$\begin{aligned}\dot{\hat{\mathbf{r}}}_r^{(3)} &= \mathbf{r}_d^{(4)} - \lambda_1\left(\hat{\mathbf{r}}^{(3)} - \mathbf{r}_d^{(3)}\right) - \lambda_2\left(\ddot{\mathbf{r}} - \ddot{\mathbf{r}}_d\right) - \lambda_3\left(\dot{\mathbf{r}} - \dot{\mathbf{r}}_d\right) \\ &= \dot{\mathbf{r}}_r^{(3)} + \lambda_1\left(\hat{\mathbf{r}}^{(3)} - \mathbf{r}^{(3)}\right),\end{aligned} \tag{5.24}$$

such that when the terms with $\dot{\Gamma}$ are neglected, equations (5.22) and (5.24) give

$$\begin{aligned} \dot{\mathbf{r}}_r^{(3)} \cdot \hat{y} &= \dot{\hat{\mathbf{r}}}_r^{(3)} \cdot \hat{y} - \lambda_1 \Gamma \omega_z \tilde{\epsilon}_x \\ \dot{\mathbf{r}}_r^{(3)} \cdot \hat{x} &= \dot{\hat{\mathbf{r}}}_r^{(3)} \cdot \hat{x} - \lambda_1 \Gamma \omega_z \tilde{\epsilon}_y. \end{aligned} \tag{5.25}$$

Expressing the body torque as the commanded torque τ_c and the unknown offset τ_o, $\tau = \tau_c + \tau_o$, we propose the following control law:

$$\begin{aligned} \tau_c &= \hat{\tau}_o - \begin{bmatrix} \Gamma^{-1}\mathbf{r}_r^{(3)} \cdot \hat{y} - \omega_z \hat{\epsilon}_x \\ -\Gamma^{-1}\mathbf{r}_r^{(3)} \cdot \hat{x} + \omega_z \hat{\epsilon}_y \\ 0 \end{bmatrix} \times J\omega + \mathbf{\Gamma^{-1}J} \begin{bmatrix} -\left(\mathbf{r}_r^{(3)} \cdot \dot{\hat{y}} + \dot{\hat{\mathbf{r}}}_r^{(3)} \cdot \hat{y}\right) \\ \left(\mathbf{r}_r^{(3)} \cdot \dot{\hat{x}} + \dot{\hat{\mathbf{r}}}_r^{(3)} \cdot \hat{x}\right) \\ 0 \end{bmatrix} \\ &\quad + J \begin{bmatrix} \hat{\epsilon}_x \dot{\omega}_z + \dot{\hat{\epsilon}}_x \omega_z \\ -\hat{\epsilon}_y \dot{\omega}_z - \dot{\hat{\epsilon}}_y \omega_z \\ 0 \end{bmatrix} - K_\tau \hat{\mathbf{S}}_\tau. \end{aligned} \tag{5.26}$$

It can be shown that using the proposed control law and equations (5.13), (5.23), and (5.25), the time derivative of the re-defined composite variable is

$$J\dot{\hat{\mathbf{S}}}_\tau = -K_\tau \hat{\mathbf{S}}_\tau + \left(\tilde{\tau}_o + Y_\tau \tilde{\mathbf{a}}\right), \tag{5.27}$$

where

$$Y_\Gamma = \begin{bmatrix} 0 & \lambda_1 J_x \omega_z & 0 \\ 0 & 0 & -\lambda_1 J_y \omega_z \\ 0 & 0 & 0 \end{bmatrix}. \tag{5.28}$$

Equation (5.27) is affine in the estimation errors $\tilde{\tau}_o$ and $\tilde{\mathbf{a}}$, rendering it possible to

apply an adaptive algorithm later. Observe that this proposed commanded torque τ_c only contains measurable variables and the adaptive parameters ($\dot{\hat{\epsilon}}_x$ and $\dot{\hat{\epsilon}}_y$) that will be given in section 5.2.3.4.

5.2.3.3 Predictor

Prior to presenting the adaptive algorithm, we first design a predictor. The idea is that some parameter errors are reflected in prediction errors. This information could be used in conjunction with the regular tracking error to estimate the unknown parameters. This strategy is generally known as composite adaptation [80]. In our case, the predictor also has a vital role in the stability property of the Lyapunov function candidate shown in the next section.

First, let s be a Laplace variable, we define a first-order low-pass filter function $f_\gamma(\cdot) = \gamma\left(s+\gamma\right)^{-1}$. The generated thrust, therefore, can be written in the form

$$\Gamma = f_\gamma\left(T_c - T_o\right) = T_f - T_o,$$

where $\Gamma_f = f_\gamma\left(T_c\right)$. The translational dynamics of the robot in equation (5.16) then becomes

$$\begin{aligned} \ddot{\mathbf{r}} &= \mathbf{g} + \left(T_f - T_o\right)\left(\hat{z} + \epsilon_x\hat{x} - \epsilon_y\hat{y}\right) \\ \ddot{\mathbf{r}} - T_f\hat{z} - \mathbf{g} &= \epsilon_x T_f\hat{x} - \epsilon_y T_f\hat{y} - T_o\hat{z} + \mathcal{O}\left(\epsilon T_o\right). \end{aligned} \tag{5.29}$$

By neglecting the second-order effects, we can apply the low-pass filter throughout twice and express the quantity on the right hand side of equation (5.29) in vector

form as

$$\begin{aligned} f_\gamma^2\left(\ddot{\mathbf{r}} - T_f\hat{z} - \mathbf{g}\right) &\approx \begin{bmatrix} -f_\gamma^2\left(\hat{z}^T\right) \\ f_\gamma^2\left(T_f\hat{x}^T\right) \\ -f_\gamma^2\left(T_f\hat{y}^T\right) \end{bmatrix}^T \mathbf{a} \\ &= W\mathbf{a}. \end{aligned} \tag{5.30}$$

At this point, we can substitute the vector $\mathbf{a}$ by its estimate $\hat{\mathbf{a}}$ and the estimation error $\tilde{\mathbf{a}}$ and rearrange the terms so that all measurable and known quantities are on the left hand side of the equation and call it ε,

$$\begin{aligned} f_\gamma^2\left(\ddot{\mathbf{r}} - T_f\hat{z} - \mathbf{g}\right) + W\hat{\mathbf{a}} &= W\tilde{\mathbf{a}} \\ \varepsilon &= W\tilde{\mathbf{a}}. \end{aligned} \tag{5.31}$$

Note that the matrix W is also measurable in real time.

5.2.3.4 Lyapunov Analysis

Here we propose a single Lyapunov function candidate for flight control of the flapping-wing MAV:

$$V = \frac{1}{2}S_\Gamma^2 + \frac{1}{2}\hat{\mathbf{S}}_\tau J\hat{\mathbf{S}}_\tau + \frac{1}{2}{}^T\tilde{\mathbf{a}}^T\Upsilon^{-1}\tilde{\mathbf{a}} + \frac{1}{2}\tilde{\tau}_o^T\Psi^{-1}\tilde{\tau}_o. \tag{5.32}$$

The first two terms in equation (5.32) correspond to altitude control and lateral position control, while the latter two terms are penalty terms for errors in the estimation of unknown parameters. Υ and Ψ are positive diagonal matrices acting as adaptive

gains.

Using the control laws presented in the preceding sections, the derivative of the Lyapunov function candidate is obtained by substituting in the results from equations (5.21) and (5.27),

$$\begin{aligned}\dot{V} &= -K_\Gamma S_\Gamma^2 - \hat{\mathbf{S}}_\tau^T K_\tau \hat{\mathbf{S}}_\tau + S_\Gamma Y_\Gamma \tilde{\mathbf{a}} + \hat{\mathbf{S}}_\tau^T \left(\tilde{\tau}_o + Y_\tau \tilde{\mathbf{a}}\right) \\ &\quad + \gamma \tilde{T}_o \left(-R_{31}\tilde{\epsilon}_x + R_{32}\tilde{\epsilon}_y\right) \\ &\quad + \dot{\hat{\mathbf{a}}}^T \Upsilon^{-1} \tilde{\mathbf{a}} + \dot{\hat{\tau}}_o^T \Psi^{-1} \tilde{\tau}_o. \end{aligned} \tag{5.33}$$

Hence, we obtain the adaptive law for the unknown torque offset,

$$\dot{\hat{\tau}}_o = -\Psi \hat{\mathbf{S}}_\tau.$$

For the estimation of $\mathbf{a}$, we propose the following adaptive algorithm:

$$\begin{aligned}\dot{\hat{\mathbf{a}}} &= -\Upsilon \left(Y_\Gamma S_\Gamma + Y_\tau \hat{\mathbf{S}}_\tau\right) \\ &\quad -\Upsilon \left(\Delta + \Sigma\right) \left(W^T W\right)^{-1} W^T \varepsilon, \end{aligned} \tag{5.34}$$

where Δ is a positive diagonal matrix and Σ is a matrix with zero diagonal elements:

$$\Sigma = \frac{\gamma}{2} \begin{bmatrix} 0 & -R_{31} & 0 \\ -R_{31} & 0 & R_{32} \\ 0 & R_{32} & 0 \end{bmatrix}. \tag{5.35}$$

This incorporation of Σ enables us to cancel out terms that are the product of two

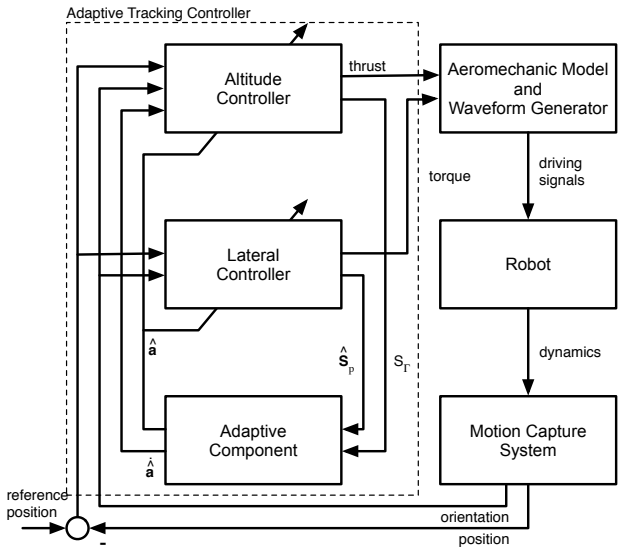

Figure 5.3: A block diagram showing the details of the adaptive controller and complete feedback loop.

parameter errors. Invertibility of W^TW depends on the rank condition of W. From equation (5.30), it can be seen that W always has full rank before being filtered. This might not be true after filtering. However, physically W is unlikely to be ill-conditioned. In practice W^TW was always found to be invertible. Lastly, substitution of equations (5.31), (5.34), and (5.35) into (5.33) yields

$$\dot{V} = -K_\Gamma S_\Gamma^2 - \hat{\mathbf{S}}_\tau^T K_\tau \hat{\mathbf{S}}_\tau - \tilde{\mathbf{a}}^T \Delta \tilde{\mathbf{a}} \leq 0, \tag{5.36}$$

that is, the derivative of the Lyapunov function candidate is negative definite. To finalize the stability proof, the invariant set theorem is applied. The value of V keeps diminishing as long as S_Γ, $\hat{\mathbf{S}}_\tau$, and $\tilde{\mathbf{a}}$ are not all zeros. The fact that $\hat{\mathbf{S}}_\tau$ approaches zero does not immediately imply that the lateral dynamics would be stabilized since when $\hat{\mathbf{S}}_\tau$ was defined in equation (5.23), it includes $\hat{\epsilon}_x$ and $\hat{\epsilon}_y$ rather than their true

values. It is the inclusion of information from the predictor that results in the last term of equation (5.36) which ensures that the parameter estimates converge to their true values, and hence $\hat{\mathbf{S}}_\tau$ eventually approaches $\mathbf{S}_\tau$ and lateral stability is satisfied along with altitude stability. The block diagram representing the relationship between various components of the complete controller is shown in figure 5.3.

5.3 Trajectory Generation

In smooth trajectory planning of robot manipulators, it is common to minimize the average squared jerk along the trajectory considered. This is in accordance with findings in psychophysical experiments on human arm movements [30]. In path planning of non-holonomic MAVs similar to our robot or quadrotors, minimum squared snap is preferred as the fourth order derivative in position can be related to torque. Hence, the optimization becomes the problem of minimizing some function of effort or torque inputs [59].

In this chapter, we devise an algorithm similar to those found in [59, 1]. The optimization routine computes a smooth polynomial trajectory from specified waypoints by minimizing feedforward torque inputs while keeping the first four derivatives of the position at the starting and ending points zero. The derivatives of the position at intermediate points are left unconstrained. The generated trajectories appear similar to those generated by minimizing the average squared snap along the trajectories.

5.4 Experiments

Without onboard sensors and control, flight control experiments are carried out in a flight arena equipped with eight motion capture *VICON* cameras as detailed in Chapter 2. Without direct measurements, velocity, acceleration, and angular velocity are not available. These quantities are estimated from the use of filtered derivatives, resulting in a slight delay to these measurements as quantified in Chapter (4).

Prior to performing unconstrained flight experiments, the robot prototype underwent a characterization process. This began with visual inspection of the flapping amplitude at various frequencies to identify a suitable operating frequency, where flapping trajectories of both wings are large and most symmetrical. Next, several open-loop takeoff flights–or *trimming* flights–were carried out in the flight arena. This process allowed us to preliminarily identify inherent torque offsets. These torque offsets vary considerably from robot to robot as a result of manufacturing imperfections. Information obtained from these steps is sufficient to perform closed-loop experiments.

5.4.1 Hovering Flight

To verify that the proposed controller is capable of stabilizing the robot, we first command the robot to hover at a stationary setpoint. After several flights, the adaptive components tuned the parameter estimates sufficiently close to their true values. In the absence of mechanical fatigue, the robot regularly stayed close to the setpoint with a position error smaller than one body length. The *Root Mean Square* (RMS) errors in position of example hovering flights are found to be comparable to those obtained from the adaptive controller in Chapter 4. The boxplots illustrating the

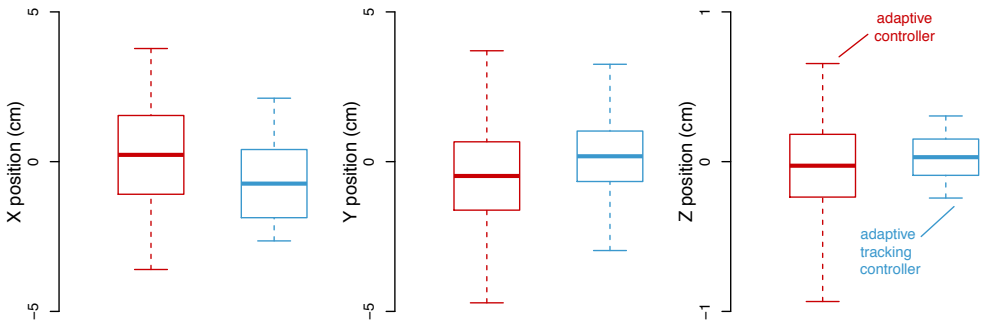

Figure 5.4: Boxplots showing the averages and the standard deviations in positions of the robots from several hovering flights using the controller derived from Chapter 4 (115 seconds or 13,800 wingbeats) and the controller proposed in the current chapter (23.5 seconds or 2,820 wingbeats).

averages and the RMS errors are shown in figure 5.4. The results shown here are consistent with the expectation that the tracking controller may not shown any significant improvement for hovering flight. Theoretically, the robot is able to stay aloft indefinitely without crashing. In practice, we attempt to minimize the total operating time to prevent mechanical fatigue.

5.4.2 Trajectory Following

To demonstrate the tracking ability of the proposed single-loop controller, we demonstrate our flapping-wing robot following a smooth trajectory generated by the algorithm described in section 6.3. The aim here is to compare the trajectory tracking performance of the adaptive tracking controller derived in this chapter and the adaptive controller presented in Chapter 4.

The trajectory in the following experiment were generated from three setpoints. The robot was set to initially hover at the starting position, navigate to the middle

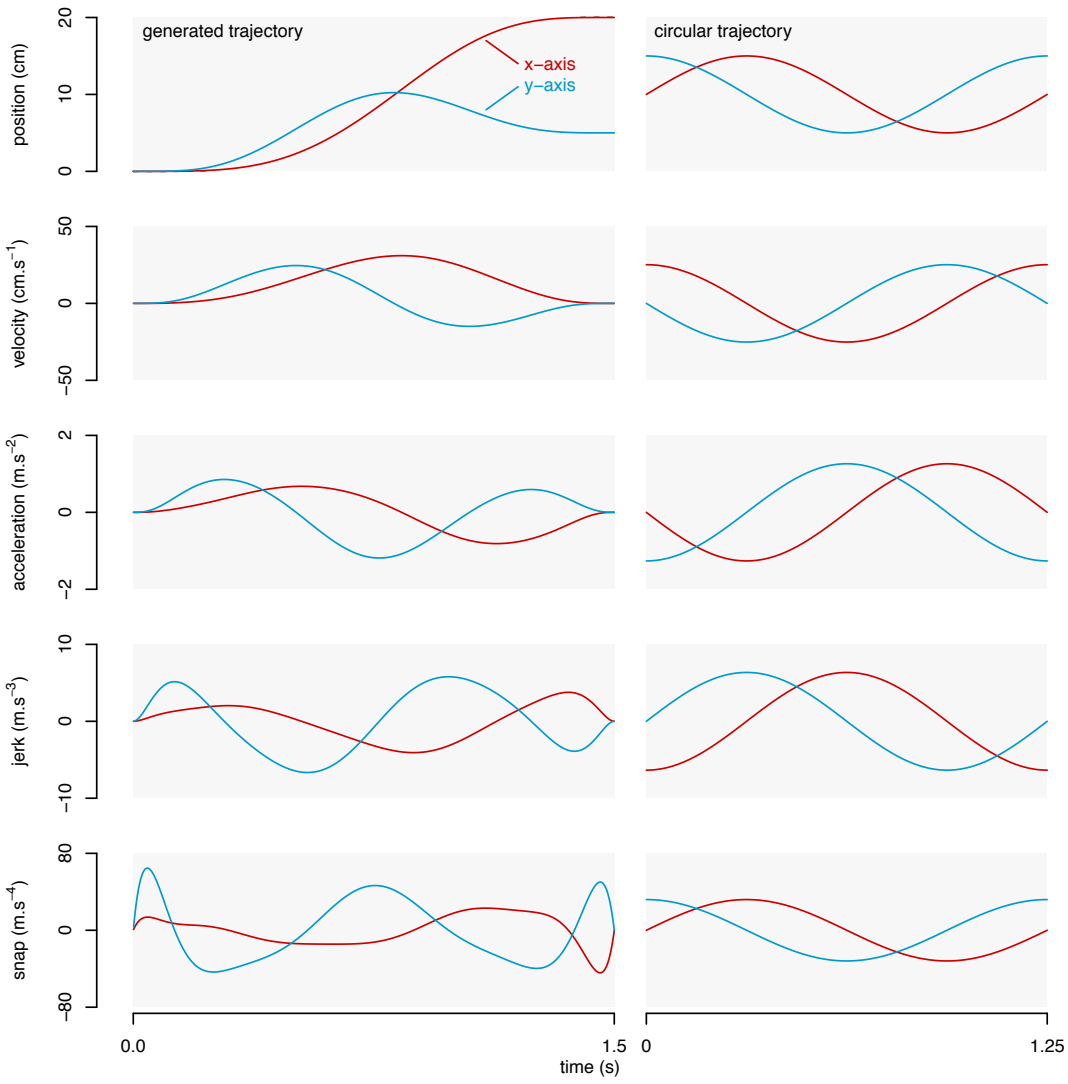

Figure 5.5: The trajectory generated for the experiments is shown on the left. Its derivatives are comparable to a more general circular trajectory with the radius and period of 5.0 cm and 1.25 s.

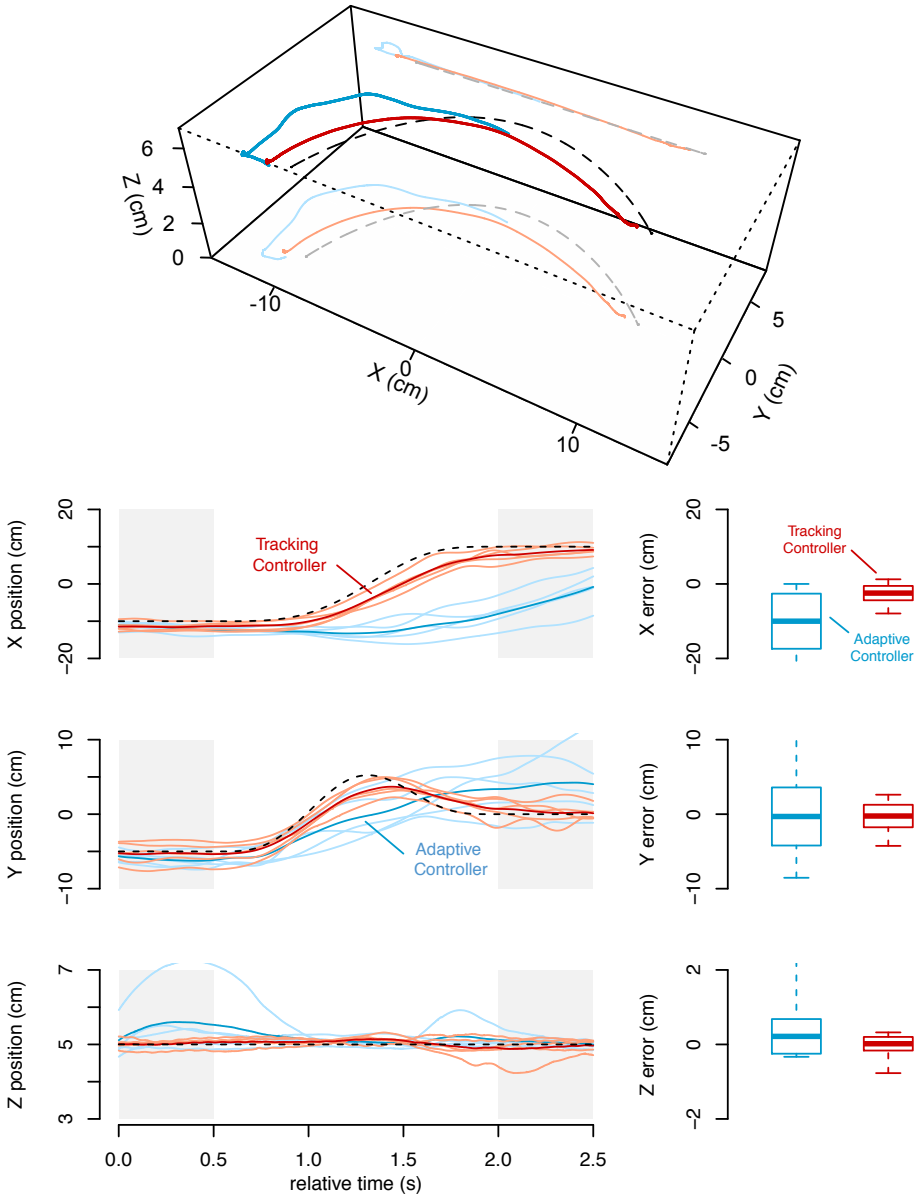

Figure 5.6: Trajectory following flights obtained from using the adaptive controller and the proposed adaptive tracking controller. (top) A 3D reconstruction of averaged flight paths from both controllers. (bottom) Plots of the position of the robots projected on to the inertial frame, compared with the reference trajectory. Solid color lines represent the average trajectories from each controller, while light-colored lines represent individual flights. Boxplots showing the average errors and standard errors along three directions are shown on the right.

point at specified times, and come to stop at the final waypoint in 1.5 seconds. First to fourth order derivatives of the position at the starting point and ending point were set to zero as detailed in the previous section. The generated trajectory, along with its respective velocity, acceleration, jerk, and snap are plotted on the left of figure 5.5. This trajectory is selected to represent general trajectories with similar properties. For instance, as shown on the right of figure 5.5, a circular trajectory with the radius of 5.0 cm and the period of 1.25 s appears to have comparable velocity, acceleration, jerk, and snap. The generated trajectory is chosen for the experiments as their first four derivatives are zero according to the imposed constraints. Moreover, the tether wire could interfere with the flight dynamics in the case of prolonged cyclic trajectories. We performed trajectories following experiments on this generated trajectory. Five flights were obtained from each controller.

Figures 5.6 show the reference trajectory and recorded paths of the robot flying through the waypoints in 1.5 seconds. Two controllers were implemented: Five flights were obtained using the adaptive controller presented in Chapter 4 and five flights were performed under the proposed tracking controller. The RMS position errors for these flights are also plotted alongside. It can be seen that the adaptive controller offers significant improvements over the adaptive controller and the RMS errors in positions are markedly smaller in all directions. This verifies that the feedforward component in the tracking controller enables the robot to follow more aggressive trajectories with greater precision.

5.5 Discussion

Towards the goal of aggressive maneuvers such as perching or acrobatic movements as demonstrated by other MAVs [60, 16, 56], we have developed a single-loop tracking controller that enables an insect-scale flapping-wing robot to follow dynamic trajectories with small errors. The single-loop design eliminates a few assumptions required in the previous controller [9] and significantly improves the tracking ability. We have demonstrated that the approach brought about improved trajectory following while retaining an adaptive ability without compromising stability.

However, it is noticeable that position errors from lateral maneuvers are still larger than those from steady hovering flights. This is unsurprising as additional aerodynamic effects were not incorporated into the dynamic model due to the lack of a simple and accurate model. It is conceivable that an improved dynamic model could contribute towards the goal of achieving more aggressive movements. This topic is covered later in Chapter 7. Alternatively, iterative learning techniques as illustrated in [60, 56] would also allow the robot to iteratively adapt the model based on information obtained from previous flights and eventually succeed in realizing pre-calculated trajectories with relatively small errors as presented in Chapter 6.

Chapter 6

Iterative Learning Control for Perching on a Vertical Surface

Inspired by the aerial prowess of flying insects, we demonstrate that their robotic counterpart, an insect-scale flapping-wing robot, can mimic an aggressive maneuver seen in natural fliers—landing on a vertical wall. Such acrobatic movement differs from simple lateral maneuvers or hover, and therefore requires additional considerations in the control strategy. Magnetic force was chosen to enable attachment to the vertical surface due to its simplicity. We show that by learning from previous failed attempts, the robotic fly could successfully perch on a magnetic wall after eight iterations.

Figure 6.1: A robot attached to a magnetic wall via the aid of 6-mil steel shims attached to the base of the landing gear.

6.1 Introduction

Over the last few decades, the flight of insects has inspired scientists and engineers to understand and translate this ubiquitous form of locomotion into man-made machines. Flies are a convenient model organism for studying insect flight—evolving sophisticated flight mechanics and exhibiting exceptional agility and maneuverability. Researchers have developed a number of biologically-inspired, flapping-wing flying vehicles [14, 71], including an insect-scale robotic fly that successfully demonstrated unconstrained but tethered flight as presented in the preceding chapters.

To date, artificial flapping wing flight at low Reynolds numbers usually relies on passive stability to achieve hover [14, 71]. Unlike [14, 71], flying insects and the robotic fly in this thesis are inherently unstable. This instability necessitates active control, but also leads to increased maneuverability. Thus far, the robotic fly has only performed basic flight maneuvers and stable hovering, and has yet to demonstrate any aggressive or acrobatic maneuvers, encountering issues in control, fast dynamics, and lack of understanding in the small-scale unsteady aerodynamics.

Other classes of Micro Aerial Vehicles (MAVs) such as quadrotors and helicopters have demonstrated highly aggressive maneuvers such as flying through narrow vertical gaps [60], performing multiflips [56], and inverted flight [64]. In these examples, a common theme in control methods is "learning". In [64], reinforcement learning was used with information from a human pilot's commands to design a controller for inverted helicopter flight. In [60, 56], iterative learning approaches were taken. It is anticipated that similar strategies could be employed for enabling aggressive aerial maneuvers with a flapping-wing flying robot.

In this chapter, we address the challenge of performing an aggressive flight maneuver with the robotic fly. Specifically, our objective is to design a flight controller and a simple attachment mechanism to allow the robot to land, or perch, on a vertical surface as illustrated in figure 6.1.

The topic of perching an MAV has been addressed previously at larger scales [12, 16, 51]. In [12], the authors placed focus on identifying an accurate model of the dynamics and utilized a value iteration algorithm in the design of the optimal control policy. Both [16, 51], on the other hand, emphasized on the design of novel attachment mechanisms that allowed the MAVs to perch within a large flight envelope.

The very small scale and payload capacity of the robotic fly in this study renders elaborate perching mechanisms as infeasible options. Fortunately, the use of magnetic force becomes more favorable at smaller scales. As length scale decreases, weight decreases as a cubic function of the characteristic length, L^3 while magnetic force scales as a function of surface area, L^2. Essentially, magnetic forces dominate gravitational forces at the small scale and when the distances are small (compared, for example, to

the characteristic dimension of the object). To exploit this, we attached small steel discs on the robot to enable the robot to land on a vertical magnetic surface.

The *Iterative Learning Control* (ILC) [6] technique was used in addition to the existing feedback control loop. The ILC algorithm allows the robot to learn from its previous flights and improves its flight performance through repetition of the same trajectory.

In section 6.2, we briefly discuss the properties of the robot, the control strategies, and the dynamic model of the perching flight. We then present a method of generating a perching trajectory followed by the formulation on the proposed ILC method in section 6.3 and 6.4. Experimental results are shown in section 6.5 followed by discussion and conclusion.

6.2 Control Strategies

6.2.1 Control Strategies

Similar to its insect counterparts, the flapping-wing robot in figure 6.1 is inherently unstable and requires active feedback control [75]. We have demonstrated that, even with advances in the manufacturing process, there are still uncertainties in the system and noticeable variations from robot to robot. In Chapter 4, it is shown that a controller able to identify and correct for some unknown parameters can improve the flight performance by reducing the position errors in hovering flights by approximately 50% (to less than 1 cm).

Achieving a stable hovering flight at this scale is challenging for a number of

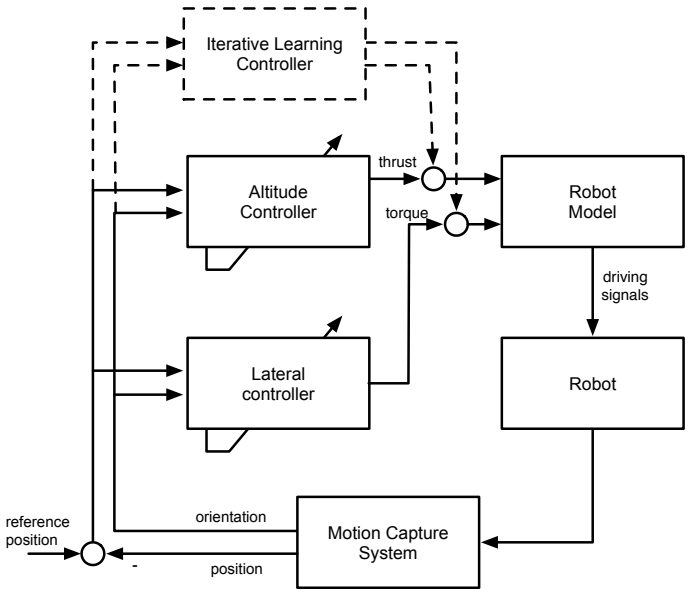

Figure 6.2: A block diagram illustrating the feedback control architecture of the robot. The ILC control block (in dashed lines) is implemented in additonal to the existing closed-loop system.

reasons: inherent vehicle instability, extremely fast dynamics, and high susceptibility to disturbances. From a controls perspective, hovering flight is a simplified case of more general maneuvers with no feedforward components. In order to realize a series of rapid maneuvers, another controller was designed in Chapter 5. This adaptive tracking controller eliminates the assumption that the attitude dynamics are generally much faster than the lateral dynamics as typically assumed in MAV literature [14, 60], while retaining an adaptive ability that was found to be crucial for flight at this scale.

In this chapter, we employ this adaptive tracking controller for preliminary trajectory following. The simplified block diagram is shown in figure 6.2. Since the dynamic model of the robot assumed by the controller is not sufficiently accurate to capture high-frequency dynamics, additional steps are required to realize more aggressive maneuvers such as perching on a vertical wall. The approach we take here is iterative

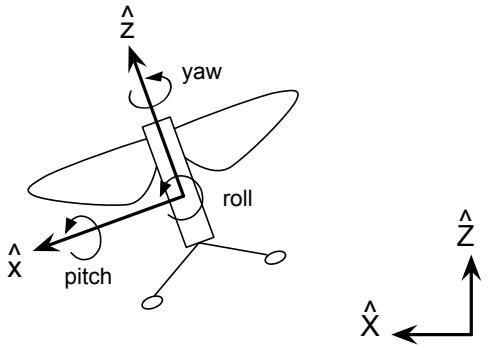

Figure 6.3: Definitions of the inertial frame, the body-attached frame, and roll, pitch, and yaw axes.

learning control, which can be implemented in addition to the current closed-loop system as highlighted in figure 6.2. Feedback is executed in real time, whereas the iterative control part is updated offline between flights.

6.2.2 2D Model of Perching Flight

To simplify the problem of landing on a vertical wall, we restrict our analysis to the two dimensional plane defined as $\hat{X} - \hat{Z}$ in the inertial coordinate frame as shown in figure 6.3. The state variables of interest consist of the position and velocity of the robot along the $\hat{X}$ and $\hat{Z}$ directions, the *tilt* angle (θ) of the robot defined as the angle between the $\hat{z}$ axis and the $\hat{Z}$ axis as projected onto the $\hat{X} - \hat{Z}$ plane (with $\hat{z}$ tilted in the $-\hat{X}$ direction defined as positive), and the normalized thrust (with the dimension of acceleration) produced by the robot (Γ). The vector containing state variables is denoted as $\mathbf{X}_s$ and is given in equation (6.1).

$$\mathbf{X}_s = \begin{bmatrix} X & \dot{X} & Z & \dot{Z} & \theta & \dot{\theta} & \Gamma \end{bmatrix}^T. \tag{6.1}$$

The dynamics of θ, the angle between $\hat{z}$ and $\hat{Z}$ projected onto the $\hat{X}-\hat{Z}$ plane, are assumed to depend on the normalized projected torque $\bar{\tau}$ as $\ddot{\theta} = \bar{\tau}$ and the normalized thrust is related to the thrust input T by the first order dynamics as $\dot{\Gamma} = -\gamma\left(\Gamma - T\right)$ for a positive constant γ, identical to the assumption in chapters 4 and 5. Therefore, T and $\bar{\tau}$ are regarded as two inputs to the system:

$$\mathbf{U} = \begin{bmatrix} T & \bar{\tau} \end{bmatrix}^T. \tag{6.2}$$

The robotic fly is an under-actuated system, similar in some regards to quadrotors [60, 56]. To maneuver laterally, the robot must tilt its body so that the thrust vector takes on a lateral component. Moreover, we assume a linear damping term in the lateral dynamics. As a consequence, the time derivative of the state vector $\mathbf{X}$ can be found from the following expression:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} \dot{X} \\ \dot{Z} \\ \dot{\theta} \\ \Gamma \end{bmatrix} = \begin{bmatrix} -\Gamma\sin\theta - \xi\dot{X} \\ \Gamma\cos\theta - g \\ \bar{\tau} \\ -\gamma\left(\Gamma - T\right) \end{bmatrix}, \tag{6.3}$$

where ξ is a damping coefficient and g is the gravitational constant. This equation formulates a framework for the analysis of trajectory generation and ILC in Sections 6.3 and 6.4.

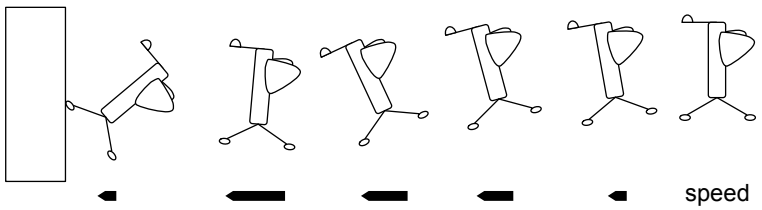

Figure 6.4: A schematic diagram demonstrating how the robot can perch on a vertical wall. Initially it needs to build up sufficient forward momentum to retain lateral velocity while rotating for perching.

6.3 Trajectory Generation

One requirement for perching on a vertical surface is to find a plausible trajectory that satisfies the constraints imposed by the dynamics of the robot as given in equation (6.3). To land on a vertical surface, there are some specifications on the trajectory, particularly at the end of the trajectory. The robot has to come to contact with the wall at steep tilt angle (preferably larger than 45°). Hence, it must generate a significant amount of torque at the very end of the trajectory. Since the torque generated is coupled with the thrust, this would decelerate the robot and potentially causes it to move away from the wall at the same time. Consequently, to perch on a wall, the robot has to carry sufficient forward momentum to assure that the robot does not move backwards. A schematic diagram illustrating a perching trajectory is shown in figure 6.4.

Mathematically, the problem can be reformulated as an optimization problem with a quadratic cost structure. However, the true purpose of the proposed framework in this section is to find a feasible (or locally optimal) trajectory that satifies the soft constraints rather than searching for the truely optimal trajectory. Here, the cost function J is comprised of an instantaneous cost $g(\cdot)$ and a terminal cost $h(\cdot)$. These

can be expressed in term of the desired states as written in equation (6.4)

$$\begin{aligned} J &= \int g\left(\mathbf{X}_s, \mathbf{U}\right) \mathrm{d}t + h\left(\mathbf{X}_{s,T}\right) \\ &= \int \begin{bmatrix} \mathbf{X}_s & \mathbf{U} \end{bmatrix} \Lambda_g \begin{bmatrix} \mathbf{X}_s & \mathbf{U} \end{bmatrix}^T - \lambda_g \dot{X} \, \mathrm{d}t \\ &\quad + \left(\mathbf{X}_{s,T} - \mathbf{X}_{s,T}^{ref}\right)^T \Lambda_{gT} \left(\mathbf{X}_T - \mathbf{X}_{s,T}^{ref}\right) - \lambda_{gT} \dot{X}_T, \end{aligned} \tag{6.4}$$

where $\mathbf{X}_{s,T}$ and $\mathbf{X}_{s,T}^{ref}$ denote the terminal state vector and the desired terminal state vector, Λ_g, Λ_{gT}, λ_g, and λ_{gT} are diagonal matrices and scalar constants. The presence of Λ_g ensures that the robot always maintains a reasonable altitude and imposes soft constraints on the input signals. The term λ_g encourages the robot to build up a forward velocity. Similarly, the final cost enforced by Λ_{gT} and λ_{gT} influences the optimizer to search for a trajectory that ends at a desired landing position and orientation with some final forward velocity.

A common practice for such optimization problems is to confine the search space. In this circumstance, we limit the inputs to be polynomial functions of time as:

$$\Gamma = \left(\sum_{i=0}^{i=N_\Gamma} a_i t^i \right)^2 \quad \bar{\tau} = \Gamma \sum_{i=0}^{i=N_\tau} b_i t^i, \tag{6.5}$$

here a_i and b_i are polynomial coefficients to be searched for. The use of polynomial structure has some benefits. For instance, by constraining b_0 to zero guarantees that a robot starting in the upright orientation will have $dX/dt = d^2X/dt^2 = d^3X/dt^3 = d^4X/dt^4 = 0$ and the trajectory is smooth at the beginning. Notice the quadratic structure of the thrust which is implemented to force the thrust to always be non-

negative. Also, the presence of Γ in the expression of $\bar{\tau}$ renders the model to be more realistic as the generation of torque is coupled with the generated thrust in the flapping-wing robot.

To find a locally optimal solution of equations (6.4) and (6.5) using gradient methods, the difficulty lies on a procedure for calculating a Jacobian. Here we compute the gradient with the *Real-Time Recurrent Learning* (RTRL) method [90, 42].

To begin, we express the dynamics of the state vector as $\dot{\mathbf{X}}_s = f(\mathbf{X}_s, \mathbf{U})$. For a parameter to optimize α (which could be a_i or b_i), we have

$$\begin{aligned} \frac{\partial}{\partial \alpha}\left(\dot{\mathbf{X}}_\mathbf{s}\right) &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathbf{X}_s}{\partial \alpha}\right) &= \frac{\partial f}{\partial \mathbf{X}_s}\frac{\partial \mathbf{X}_s}{\partial \alpha} + \frac{\partial f}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \alpha} \\ &= \frac{\mathrm{d}}{\mathrm{d}t}P &= \frac{\partial f}{\partial \mathbf{X}_s}P + \frac{\partial f}{\partial \mathbf{U}}Q, \end{aligned} \tag{6.6}$$

where P and Q have been defined as $\partial \mathbf{X}_s/\partial\alpha$ and $\partial \mathbf{U}/\partial\alpha$ respectively. According to equation (6.4), the Jacobian $\partial J/\partial\alpha$ is then simply given as

$$\frac{\partial J}{\partial \alpha} = \int \left(\frac{\partial g}{\partial \mathbf{X}_s}P + \frac{\partial g}{\partial \mathbf{U}}Q\right) dt + \frac{\partial h}{\partial \mathbf{X}_s}P + \frac{\partial h}{\partial \mathbf{U}}Q. \tag{6.7}$$

It is straightforward to obtain $\partial f/\partial \mathbf{X}_s$ and $\partial f/\partial \mathbf{U}$ from equation (6.3). Thus, by integrating forward equation (6.6) to find P, the gradient $\partial J/\partial\alpha$ can be evaluated from equation (6.7). To perform a gradient descent, the cost function is expanded to its second order approximation as

$$J(\alpha + \delta\alpha) \approx J(\alpha) + \left(\frac{\partial J}{\partial \alpha}\right)^T \delta\alpha + \frac{1}{2}\delta\alpha^T\left(\frac{\partial^2 J}{\partial \alpha^2}\right)\delta\alpha. \tag{6.8}$$

Inspired by the Gauss-Newton algorithm, here we opt to approximate the Hessian

by taking a derivative of equation (6.7) with respect to α again but neglecting the $\partial^2 P/\partial\alpha^2$ and $\partial^2 Q/\partial\alpha^2$ terms. This reduces the complexity and the computational time. It follows that we can then solve equation (6.8) for an incremental change in α:

$$\delta\alpha = -\eta \left(\frac{\partial^2 J}{\partial \alpha^2}\right)^{-1} \frac{\partial J}{\partial \alpha}.$$

The step size parameter η keeps the update gradual, which improves the stability. Performance could also be tweaked by altering the cost and the reference state.

6.4 Iterative Learning Control for Perching on a Vertical Surface

After each flight iteration, the recorded trajectory is analyzed for the iterative learning controller to compute a set of corrective commands as inputs for the next flight so that the flight trajectory will eventually converge to the reference trajectory. The major distinction between a closed-loop controller and the iterative learning controller is that the former does not primarily learn from prior experiences (except for the adaptive part, nevertheless, the adaptive algorithm is not time or trajectory specific). The learning controller, on the other hand, relies solely on repetition and repetitive disturbances or systematic errors in the modeling.

To consider a whole trajectory, we consider the dynamics using a lifted representation, similar to the approach taken in [76]. That is, we discretize and consolidate

the state vectors and the inputs into a long vector given by the following

$$\begin{aligned} \mathbf{X}^* &= \begin{bmatrix} \mathbf{X}_s(t_1) & \mathbf{X}_s(t_2) & \dots & \mathbf{X}_s(t_N) \end{bmatrix}^T \\ \mathbf{U}^* &= \begin{bmatrix} \mathbf{U}(t_1) & \mathbf{U}(t_2) & \dots & \mathbf{U}(t_N) \end{bmatrix}^T . \end{aligned} \tag{6.9}$$

The model of the lifted dynamics is given by a function $f^*(\cdot)$. If we assume a perfect model, then the reference trajectory $\mathbf{X}^*_{ref}$ can be realized using a feedforward input $\mathbf{U}^*_{ff}$ as

$$\dot{\mathbf{X}}^*_{ref} = f^*\left(\mathbf{U}^*_{ff}\right). \tag{6.10}$$

In reality, it is not anticipated that the model will be perfect. Instead of attempting to find a better model, we assume that the input into the system can be regarded as a combination of the command input and the unknown disturbance input $\mathbf{U}^* = \mathbf{U}^*_c - \mathbf{U}^*_d$, and the ultimate goal of the algorithm is to find the estimate of the unknown disturbance input $\hat{\mathbf{U}}^*_d$. When we have an accurate estimate of the unknown disturbance input, we can achieve the reference trajectory by using the command input $\mathbf{U}^*_c = \mathbf{U}^*_{ff} + \hat{\mathbf{U}}^*_d$ as given below:

$$\begin{aligned} \dot{\mathbf{X}}^* &= f^*(\mathbf{U}^*) \\ &= f^*(\mathbf{U}^*_c - \mathbf{U}^*_d) \\ &= f^*\left(\mathbf{U}^*_{ff} + \hat{\mathbf{U}}^*_d - \mathbf{U}^*_d\right). \end{aligned} \tag{6.11}$$

In order to calculate the unknown disturbance input, we first define the estimation

error at iteration j as

$$\tilde{\mathbf{U}}^*_{d,j} = \hat{\mathbf{U}}^*_{d,j} - \mathbf{U}^*_d. \tag{6.12}$$

Then the lifted dynamics equation can be expanded about the ideal operating point $\mathbf{U}^*_{ff}$,

$$\dot{\mathbf{X}}^*_j \approx f^*\left(\mathbf{U}^*_{ff}\right) + \left(\frac{\mathrm{d}}{\mathrm{d}\mathbf{U}^*} f^*\Big|_{\mathbf{U}^*_{ff}}\right) \cdot \tilde{\mathbf{U}}^*_{d,j}. \tag{6.13}$$

The quantity on the left hand side of equation (6.13) could be obtained by post-processing the recorded trajectory. The difference between the measured $\dot{\mathbf{X}}^*_j$ and $\dot{\mathbf{X}}^*_{ref}$ forms an error vector $\mathbf{e}_j$ that is a function of $\tilde{\mathbf{U}}^*_{d,j}$

$$\mathbf{e}_j = \dot{\mathbf{X}}^*_j - \dot{\mathbf{X}}^*_{ref} = \left(\frac{\mathrm{d}}{\mathrm{d}\mathbf{U}^*} f^*\Big|_{\mathbf{U}^*_{ff}}\right) \tilde{\mathbf{U}}^*_{d,j} = F\tilde{\mathbf{U}}^*_{d,j}.$$

It can be seen that the matrix F is only a function of the reference trajectory and is independent of the current trajectory, so it only needs to be computed once. At this point, we propose an update law for the estimate of $\mathbf{U}^*_d$ for the next iteration:

$$\hat{\mathbf{U}}^*_{d,j+1} = \hat{\mathbf{U}}^*_{d,j} - F^T \Delta \mathbf{e}_j,$$

for some positive diagonal matrix Δ. It follows that we could express the l_2-norm of the error vector from two consecutive iterations as

$$\mathbf{e}^T_{j+1}\mathbf{e}_{j+1} = \mathbf{e}^T_j \left(I - FF^T\Delta\right)^T \left(I - FF^T\Delta\right) \mathbf{e}_j.$$

Since FF^T is always positive definite, for a sufficiently small Δ, the norm of the error vector always decreases. In other words, we have

$$\mathbf{e}_{j+1}^T \mathbf{e}_{j+1} \leq \sigma \mathbf{e}_j^T \mathbf{e}_j \qquad \text{for} \quad \exists\, 0 \leq \sigma < 1.$$

In the implementation, F can be computed by representing the robot's dynamics given by equation (6.3) along the reference trajectory using a *Linear Time Varying* (LTV) configuration.

$$\dot{\mathbf{X}}_s(t) = A(t)\mathbf{X}_s(t) + B(t)\mathbf{U}(t),$$

and the matrix F is simply given by

$$\begin{aligned} F &= \begin{bmatrix} \frac{\partial f_{t_1}}{\partial \mathbf{U}_1} & \cdots & \frac{\partial f_{t_1}}{\partial \mathbf{U}_{t_N}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{t_N}}{\partial \mathbf{U}_1} & \cdots & \frac{\partial f_{t_N}}{\partial \mathbf{U}_{t_N}} \end{bmatrix} \\ &= \begin{bmatrix} F_{(1,1)} & \cdots & F_{(1,N)} \\ \vdots & \ddots & \vdots \\ F_{(N,1)} & \cdots & F_{(N,N)} \end{bmatrix}, \end{aligned}$$

where the elements can be shown to be

$$F_{(\mathrm{m,n})} = \begin{cases} 0 & \text{if } m < n \\ B(t_n) & \text{if } m = n \\ B(t_n) \sum_{j=n+1}^{j=m} \left(\prod_{i=j}^{i=m} A(t_i) dt \right) & \text{if } m > n. \end{cases}$$

6.4.1 Consideration of Initial Conditions

In the previous section, it is shown that the norm of error vector should gradually decrease after each iteration. However, the presented analysis assumes that each trajectory always starts with the same initial conditions, in a way that minimizing the error in $\dot{\mathbf{X}}^*$ is sufficient to bring the actual trajectory close to the reference trajectory. In our case, where the initial condition involves a robot hovering in place, that condition is approximately satisfied for the tilt angle and the initial velocity, but not for the position. In the previous chapters, it was shown that the RMS of the position error during hover was just below 1 cm. This means that even if the error vector becomes zero, the robot could end up attempting to perch one centimeter away from the wall.

To avoid a situation similar to the one mentioned above, we allow the robot to initialize a perching attempt only when the attitude is stable as measured by a metric given by the controller in Chapter 5. Furthermore, when the starting position is not zero, trajectory tracking will not start from the beginning of the pre-planned trajectory. To demonstrate, suppose the robot starts perching at time t_0 with $X(t_0) = X_0$ and the reference trajectory is defined for $\mathbf{X}^{ref}(t')$ for $0 \leq t' \leq T_f$, we seek to find t'_0 that satisfies the equation

$$X_0 = X^{ref}(t'_0) + \int_{t'_0}^{T_f} \dot{X}^{ref}(t)dt,$$

and command the robot to follow the trajectory from $\mathbf{X}_s^{ref}(t'_0)$ to $\mathbf{X}_s^{ref}(T_f)$. The idea is that, to a first order approximation, the extra distance at the beginning would

eventually be cancelled out by the deficit in the initial velocity. As a result, the robot's trajectory will not match the reference at the beginning ($X(t_0) \neq X^{ref}(t'_0)$), but the discrepancy should theoretically diminish towards the end.

6.4.2 Implementation in Three Dimensions

In practice, the controller only takes into consideration the direction of the $\hat{z}$ axis of the robot and does not directly control the heading direction (yaw orientation) of the robot. In other words, the robot would need to perform both pitch and roll maneuvers to realize the reference trajectory depending on its current orientation. In the case that the robot follows a trajectory to perch on a wall in the positive $\hat{X}$ direction, the reference torque input $\bar{\tau}$ points along a negative $\hat{Y}$ direction. With the knowledge of the current orientation of the robot and the assumption that the moment of inertia along the pitch and roll axes are approximately equal, it is possible to find a combination of pitch and roll torques in the body frame that point in the negative $\hat{Y}$ direction with the specified magnitude.

6.5 Experiments and Results

6.5.1 Trajectory Optimization

Initially, the perching trajectory is crudely hand-designed with a target distance near 12 cm from the starting position and the trajectory duration of 0.65 s. This was then parametrized by polynomial functions as given by equation (6.5) with $N_\Gamma = 8$ and $N_\tau = 12$. The parametrized trajectory is illustrated with grey lines in figure

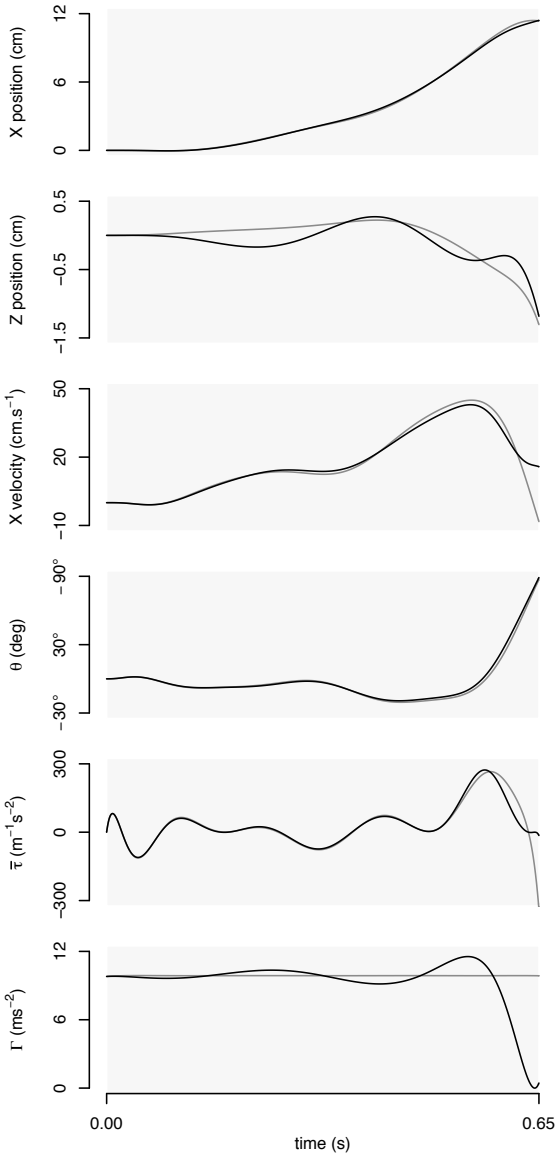

Figure 6.5: A candidate trajectory for perching. The grey lines show a parametrized version of a hand-designed trajectory. Black lines represent the final trajectory after the optimization.

6.5. Next, the trajectory is optimized using the strategy proposed in section 6.3 and the results are shown as black lines in figure 6.5. It can be seen that the most significant difference is in the terminal velocity which is, in fact, negative before the optimization. The optimized thrust is also small at the end. Understandably this is in order to reduce the amount of deceleration and preserve the forward momentum. The resultant trajectory is 11.4 cm long with the projected terminal tilt angle close to 90°.

6.5.2 Landing Mechanism

The magnetic wall was constructed from a flexible magnetic sheet manufactured from Ferrite bonded with synthetic rubber. This type of magnet is generally classifed to have very low magnetic pull. The maximum pull (defined as the maximum pull force to a thick flat steel plate in an ideal laboratory condition) is 1100 $\text{kg}\cdot\text{m}^{-2}$. On the robot, four discs of 6 mil (0.15 mm) steel shims, each with a diameter of 2 mm, were attached to the landing gear. This brought up the total weight of the robot from 80 mg to 100mg—the change that needs to be accounted for by the control algorithm.

We experimentally found that one disc of steel shim could hold a weight of up to 230 mg. In an ideal case–neglecting the force required to counter any kinetic energy–, a simple calculation reveals that one disc must be able to support at least ≈ 60 mg to hold the robot to the wall during static conditions. Taking other factors into consideration, the strength of the magnet and the size of the steel shims offer appropriate attraction for the landing task. Furthermore, the field of a magnetic sheet is expected to decay faster than that of a magnetic dipole, which is an inverse cubic

function of distance, or r^{-3} [45]. As a result, the contribution of the magnetic force should be negligible when the robot is not in contact with the magnetic wall.

6.5.3 Experimental Results

Prior to perching experiments, the robot has to be verified for its flight capability. This involves the characterization of the robot's flapping amplitude at various operating frequencies. After validating that the robot possesses sufficiently large and symmetrical flapping amplitudes on both wings, the robot needs to be trimmed for flight. The trimming process starts with short, unstable open-loop flights that each lasts less than 0.4 s to determine a set of driving signals that minimize the residual torque exhibited by the robot, due to unavoidable mechanical asymmetries. These are followed by a closed-loop trimming process, in which the adaptive part of the controller corrects for torque offsets further until the robot can hover with position errors on the order of 1cm or less.

We must ensure the robot will start its trajectory on or somewhere in front of the prescribed perching trajectory's starting point. Because of this one-centimeter uncertainty in the starting position of the robot, we actually define the start of the trajectory to be at -1.0 cm from the hovering setpoint of the robot at 0.0 cm. The target vertical wall is placed at 10.4 cm from the hovering setpoint. The controller allowed the perching attempt to begin only when the robot is less than one centimeter away from the setpoint (-1.0 cm $\leq X \leq 1.0$ cm). Thus, given the 11.4cm prescribed trajectory, the robot will start with an initial condition lying in front of the trajectory's starting point.

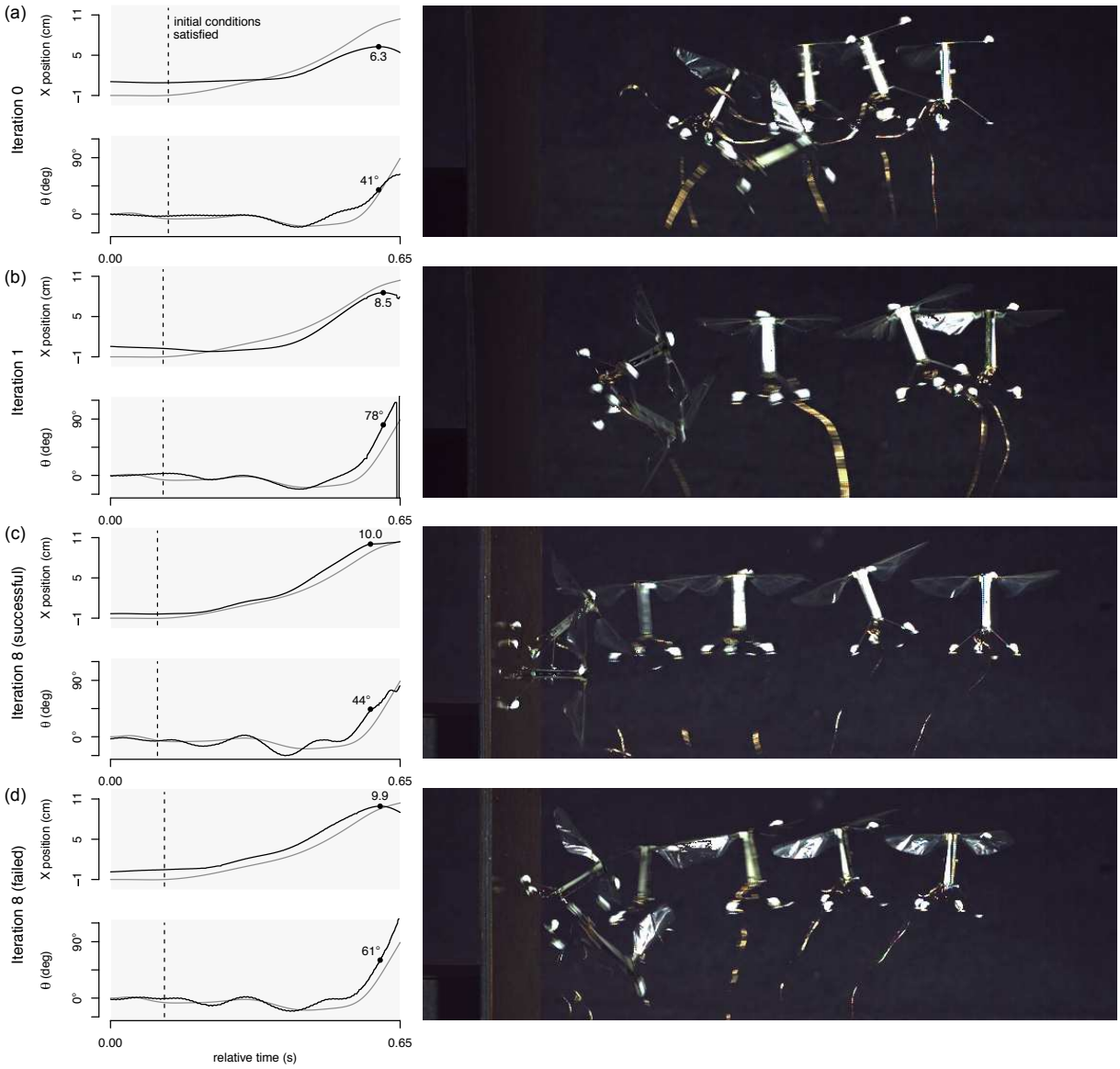

Figure 6.6: Recorded trajectories accompanied by composite images constructed from the videos. References are shown in grey lines and the actual trajectories are in black lines. (a) The trajectory obtained without any command from the ILC algorithm. (b) The trajectory generated after one learning iteration. (c) A successful trajectory obtained after eight iterations of learning. (d) A failed perching attempt obtained from the same command as in (c).

The first perching flight (iteration 0) was executed with no correction from the ILC algorithm. The plots of this iteration's trajectory are illustrated in figure 6.6(a). The robot only reaches a distance of 6.3 cm and a tilt angle of 41° before falling out of the air.

In iteration 1, after implementing the first estimate of $\mathbf{U}_d^*$, the robot could get closer to the wall, i.e., it achieved the distance of 8.5cm before losing all the forward momentum, at which point the robot had a tilt angle of 78°. The corresponding trajectory is shown in figure 6.6(b).

Due to the nature of the experiments and the delicate character of the robot, it is impossible to ensure that the physical properties of the robot remain unaltered over the course of several flights. Sources of uncertainty include mechanical fatigue of the wing hinges, wings, and actuators (which unfortunately do occur over the timescale of the experiments), structural damage from crashing to the ground, and electrical connection failure due to wire fatigue. After subsequent repairs to the robot, flight trimming experiments must be repeated—allowing the adaptive part of the controller to trim the robot for a good operating condition.

On this occasion, both wing hinges on the robot mechanically failed after six iterations just as the robot was close to achieving a successful perching trajectory. Both wings and hinges were replaced and the trimming process was carried out again to achieve a steady hover. It is noted that the operating point of the repaired robot was different than that of the robot prior to wing hinge failure.

Using the same command resulted in slightly different trajectories compared with those obtained before wing hinge failure. Nevertheless, we carried on applying the

ILC algorithm and updated the estimate of $\mathbf{U}_d^*$ from the previous iteration instead of resetting the iterative process. After two subsequent iterations (iteration 8), the robot successfully landed on the vertical wall. A few subsequent flights using the same command resulted in a mix of successful and failed wall perchings. One example of a successful attempt is demonstrated in figure 6.6(c). In this attempt, the robot first contacted the wall when the tilt angle was around 45°. A failed perching attempt is also presented in figure 6.6(d). The recorded trajectory reveals that the robot missed the target setpoint by 1-2mm. This also suggests that the effect of the magnetic force is minimal when the robot is not attached to the wall.

6.6 Discussion and Conclusion

In this chapter, we have shown that a millimeter-scale, flapping-wing flying robot is capable of performing an aggressive aerial maneuver—perching on a vertical wall. From a controls perspective, such a maneuver differs considerably from hover or slow maneuvers that have been demonstrated in previous chapters. To land on a vertical surface, first we constructed a nominal perching trajectory via an optimization method that assumes a simplified dynamics model in two dimensions. For control, we opted to implement an iterative learning control algorithm in addition to the existing adaptive tracking flight controller from chapter 5. This learning algorithm computed an updated feedforward command for the robot after each perching attempt, in order to improve the trajectory tracking performance in an iterative fashion. Due to stringent payload constraints and scalability challenges, magnetic force was utilized as the wall attachment mechanism, enabling the robot to perch on a vertical magnetic

surface. The magnetic force is only sufficient to hold the robot when it makes surface contact and has an insignificant effect on broadening the flight trajectory envelope for successful wall perches. It is shown that after eight iterations, the proposed control strategies enabled the robot to successfully land on a vertical surface as desired.

Without the learning algorithm used in this chapter, the existing controller is unable to command the robot in following the prescribed trajectory. Understandably, like most controllers, the dynamic model assumed by the previous controller only accurately captures slow system dynamics, lacking fidelity for unmodeled high frequency components required to perform an aggressive maneuver. Yet, this nominal model is sufficiently accurate to form a basis for the search for a feasible trajectory and the learning process to compensate for the inaccuracies by repeating the trajectory following attempts taking into consideration only the first-order approximations of the dynamics. We show that the robot is able to land on a vertical surface—a task that requires millimeter-accuracy for a robot in which the position error of its stationary hovering flight is in the range of one centimeter.

The utilization of magnetic force is convenient for a robot at this scale because its implementation adds minimal payload. Unfortunately, while it is sufficient to enable demonstrations of aggressive trajectory-following, it does not allow the robot to autonomously takeoff from the wall's surface. A more finely tuned or altogether different attachment mechanism is needed. It is not trivial to construct a detachable attachment mechanism similar to those seen in [40, 61], let alone at this much smaller scale. However, we predict that once a more elaborate attachment mechanism is developed, the control strategy illustrated in this chapter is suitable for direct application.

Chapter 7

System Identification of In-Flight Aerodynamics

In this chapter, we construct a physics-based model to describe the effects of aerodynamic drag that contributes to the translational and rotational dynamics of the robot. The effects, previously neglected by the controller, are identified from experimental flight data via linear regression. These identified parameters correspond to relevant physical parameters from the proposed grey-box models. The learned models, which also have some implications on the dynamics and stability of the robot, could be incorporated into the flight controller in order to improve the flight performance of the robot.

7.1 Introduction

From the first nonlinear flight controller proposed in chapter 3, we have gradually improved our understanding on the flight dynamics of the robot. This translates to progressively more sophisticated flight controllers as seen in chapters 4-6, where more considerations have been incorporated into the controllers to improve the flight performance. To this point, however, we have neglected the effects of additional aerodynamic drag acting on the robot during flight, owing to the lack of understanding, and, hence, the lack of sufficiently accurate models and corresponding parameters suitable for flight control purposes.

Regarding the topic of evaluation of unknown physical parameters, thus far, we have employed adaptive techniques to estimate some unknown parameters in an online fashion. To further improve the tracking precision, in chapter 6, the iterative learning algorithm is used to improve the tracking performance of a highly aggresive but specific trajectory. The iterative learning control technique relies on a dynamic model of the robot and previous results to calculate updated nominal inputs to the system to compensate for unmodeled dynamics and improve the flight precision on a specific trajectory.

In addition to the adaptive approach and the iterative learning method, another strategy to gain further insights into the system based on the flight data is to conduct offline parameter estimations, which is loosely classified as model identification or system identification in control and aircraft communities. A number of researchers have applied identification techniques to stable aerial vehicles to quantifies the aerodynamic effects and their responses to the actuation commands [37, 7, 50, 42]. For an

unstable system such as our robot, a closed-loop treatment is required and extra care must be taken [39, 31] to avoid undesriable coupling between the input commands and dynamic responses. One suitable approach for closed-loop identification taken in this chapter is the grey-box model approach [55]. This involves searching for parameter estimates that minimize some cost function based on a presumed model structure.

For the sake of simplicity, we opt to express our models in the form of linear predictor functions in a way that the estimates of unknown parameters could be evaluated via linear regression as regression coefficients. The decision to use linear structures brings about several beneficial aspects, including the existence of global extremum and analytic solutions, which obviate the need to tune the optimizing parameters and the need to perform a local search that could be relatively computationally expensive as carried out in [10]. More complicated model structures could be achieved while retaining a linear structure via the use of basis functions as found in [42].

In this chapter, we aim to characterize the translational and rotational dynamics of the robot, emphasizing the effects of aerodynamic drag that arises from the translation and rotation of the robot in flight. At the beginning of section 7.2, background in the origin of in-flight aerodynamic contributions is briefly covered. This is followed by its consequential effects on the translational and rotational dynamics of the robot. Then, we present how the dynamic equations can be structured into forms suitable for linear regression. The estimation of parameters from recorded flight data are carried out. Lastly, the outcomes and the implications on the stability of the robot are discussed.

7.2 In-Flight Aerodynamic Model

In [73, 75], instantaneous drag from the wings during flight was approximated by considering the instantaneous drag on the two wings by the high Reynolds number law

$$D = 2 \times \frac{1}{2}\rho S C_D v^2, \tag{7.1}$$

where ρ is the fluid density, S is the wing area, C_D is the coefficient of drag, and v is the wing speed relative to the air. When deviated from a perfect hovering condition, an additonal damping force arises from the difference between the forward drag and the backward drag from the robot moving at speed u, much smaller than the nominal mid-stroke wing speed v:

$$\begin{aligned} D &= 2 \times \frac{1}{2}\rho S C_D \left[(v+u)^2 - (v-u)^2\right] \\ &\approx 4\rho S C_D v \cdot u = mbu, \end{aligned} \tag{7.2}$$

where m is the body mass of the robot, and b is a normalized drag coefficient. This approximation ignores the second-order terms $(u/v)^2$, which is a justified assumption, given that a wing longer than one centimeter flapping at 120 Hz with a peak-to-peak amplitude of 90° easily reaches the speed up to $5 - 6$ m·s^{-1}, whereas the expected flight speed is on the order of 10 cm·s^{-1}. Equation (7.2) implies that the additional in-flight aerodynamic drag can be modeled as a linear viscous term. In general, rotations also contribute to the perceived wind speed encountering by the wings. For a robot rotating at a rate ω with the distance from the center of mass to the center of drag

(or the effective length of the moment arm) l, we may express the drag force as

$$D \approx mb\,(u + l \cdot \omega) = mbu + m\alpha\omega. \tag{7.3}$$

Furthermore, the drag force in equation (7.3) also contributes to the rotational damping. The rotational damping coefficients could be expressed as the damping force multiplied by the effective moment arm, resulting in the following drag moment:

$$M \approx m\alpha l \cdot \omega + mbl \cdot u = Jc\omega + J\beta u, \tag{7.4}$$

where c and β are corresponding rotational damping coefficients, normalized by the moment of inertia. The implication of equations (7.3) and (7.4) is that viscous forces and torques from flapping wings, which dominate the body drag, can be treated as linear functions, as opposed to quadratic functions. These viscous components were shown to be critical to open-loop dynamics and stability of similar-scaled flapping-wing insects [75, 67, 24].

7.2.1 Translational Dynamics

In Chapter 2, we first presented equation (2.5) that describes the translational dynamics of the robot in $\mathbb{R}^3$. This equation is reproduced below as equation (7.5). Thus far, it has been used to derive the flight controllers in Chapter 3-6 without taking into account the additional aerodynamic forces, which could be lumped into

the $\mathbf{F}_{add}$ term in equation (7.5).

$$\ddot{\mathbf{X}} = \Gamma\hat{z} + \mathbf{g} + \mathbf{F}_{add}. \tag{7.5}$$

In the body-fixed coordinate frame, the rotation of the frame has to be taken into consideration. Equation (7.5) can be re written as

$$\begin{aligned} \sum \mathbf{F} &= \frac{\mathrm{d}}{\mathrm{d}t}\dot{\mathbf{x}} + \omega \times \dot{\mathbf{x}} \\ &= \begin{bmatrix} 0 \\ 0 \\ \Gamma \end{bmatrix} + R^T \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \mathbf{f}_{add}. \end{aligned} \tag{7.6}$$

In this chapter, we seek to find a simple representation of $\mathbf{f}_{add}$ as a function of the state variables. It is conceivable that such additional aerodynamic forces result primarily from drag forces on acting the wings and viscous forces on the body of the robot. As discussed previously, we have shown that the flapping wings–the dominant sources of drag–give rise to drag forces and torques that are linear functions of the translational and angular velocities. The damping coefficients depend on the aerodynamic characteristics, such as size, as well as other fluid properties as outlined at the beginning of this section. In summary, these additonal forces can be approximately

written as functions of velocities as

$$\mathbf{f}_{add} = \begin{bmatrix} -b_x\dot{x} - \alpha_x\omega_y \\ -b_y\dot{y} + \alpha_y\omega_x \\ -b_z\dot{z} \end{bmatrix},$$

where b_i's and α_i's are coefficients that capture the contribution of body velocity and angular velocity. It follows that, in the body-fixed frame, the translational dynamics of the robot are described by the following equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\dot{\mathbf{x}} + \omega \times \dot{\mathbf{x}} = \begin{bmatrix} 0 \\ 0 \\ \mathrm{r} \end{bmatrix} - g \begin{bmatrix} R_{31} \\ R_{32} \\ R_{33} \end{bmatrix} + \begin{bmatrix} -b_x\dot{x} - \alpha_x\omega_y \\ -b_y\dot{y} + \alpha_y\omega_x \\ -b_z\dot{z} \end{bmatrix}. \tag{7.7}$$

7.2.2 Rotational Dynamics

Similar to the translational dynamics, the rotational dynamics are also affected by additional torques when the robot is not in a perfect hovering state. Subsequently, the resultant torque acting on the robot could be written as a combination of the nominal torque produced by flapping wings as commanded by the controller τ_c and the additional torque due to the velocity and the angular velocity of the robot τ_{add}. Equation (2.2) then becomes

$$\begin{aligned} \sum \tau &= J\frac{\mathrm{d}}{\mathrm{d}\mathbf{t}}\omega + \omega \times J\omega \\ &= \tau_c + \tau_{add} + Jd, \end{aligned} \tag{7.8}$$

where we have included an unknown normalized disturbance term Jd. In this chapter, we focus on rotation about the roll and pitch axes. The contribution from the additional aerodynamic effects are approximated as linear similar to the case of the translational dynamics. That is the drag torque can be modeled as the drag force multiplied by the effective moment arm. Therefore, the first two components of equation (7.8) can be rearranged into

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\omega_x\\ \omega_y\end{bmatrix} = \begin{bmatrix}\tau_{c,x}J_x^{-1}\\ \tau_{c,y}J_y^{-1}\end{bmatrix} + \begin{bmatrix}(J_y-J_z)\,J_x^{-1} & 0\\ 0 & (J_z-J_x)\,J_y^{-1}\end{bmatrix}\begin{bmatrix}\omega_y\omega_z\\ \omega_x\omega_z\end{bmatrix} + \begin{bmatrix}-c_x\omega_x+\beta_x\dot{y}\\ -c_y\omega_y-\beta_y\dot{x}\end{bmatrix} + \begin{bmatrix}d_x\\ d_y\end{bmatrix}, \tag{7.9}$$

where c_i's and β_i's are normalized drag coefficients that reflect the contribution from the angular velocities and the translational velocities.

7.3 Flight Trajectories and Linear Regression

Equations (7.7) and (7.9) possess linear structures that are suitable for unknown parameter estimation via linear regression. The left hand side of both equations consist of quantities that could be extracted from the recorded trajectories of the robot. This, when taken from various times and trajectories, could be consolidated as an $N \times 1$ vector Ψ, where N is the number of total observations from all relevent trajectories. The right hand side of equations (7.7) and (7.9) can be re-arranged such that all unknowns parameters construct an $n \times 1$ vector φ. Relevant state variables

are in an $N \times n$ matrix χ:

$$\begin{aligned} \Psi &= \chi\varphi + \epsilon \\ \Psi_i &= \varphi_0 + \sum_{j=1}^{n} \chi_{ij}\varphi_j + \epsilon_i \quad \text{for } i \in \{1, 2, .., N\}, \end{aligned} \tag{7.10}$$

where φ_0 is an affine coefficient that accounts for possible unknown offsets, and ϵ_i is the residual for the i^{th} observation. An ordinary least squares estimator is used to estimate the parameters in the linear models:

$$\hat{\varphi} = \left(\chi^T\chi\right)^{-1}\chi^T\Psi.$$

The residual ϵ is the difference between the observed data and the fitted model. It is indicative of the goodness of fit. One measure of the goodness of the fit is the coefficient of determination, or R^2. Statistically, R^2 indicates how closely values obtained from fitting a model match the dependent variable the model is intended to predict which can be found as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\epsilon_i^2}{\sum_{i=1}^{N}\left(\Psi_i - \frac{1}{N}\sum_{j=1}^{N}\Psi_j\right)^2}.$$

An R^2 value near unity indicates that the model $\chi\varphi$ explains most of the variability Ψ, on the other hand, an R^2 value near zero indicates that the fit is not much better than the model $\chi\varphi$ being constant.

7.3.1 Preprocessing

Prior to model fitting, recorded flight trajectories are pre-processed. The beginning portions of flights immediately after taking off, where thrust is ramping up, are discarded. The data is down-sampled to 1,000Hz using a cubic spline to enhance the continuity. This enables us to smooth the trajectories by acausally applying a 3^{rd}order Chebyshev-type-II low-pass filter to the data sequence. The cutoff frequency of the filter is chosen to be 50Hz in order to filter out the oscillations caused by the flapping wings at 120Hz.

The filtered datasets are numerically processed and differentiated. For instance, angular velocities are constructed from the rotation matrix and its derivative as given in equation (2.3), positions are differentiated and projected onto the body-fixed coordinates to obtain the velocities in the body-fixed frame.

7.3.2 Lateral Dynamics

The translational dynamics in the body-fixed frame given in equation (7.6) can be broken down into lateral dynamics and altitude dynamic. In this section, we consider the lateral dynamics along the $\hat{x}$ and $\hat{y}$ axes from equation (7.6), which can be re-organized into the structure suitable for linear regression as outlined in equation (7.10). In this circumstance, both $\hat{x}$ and $\hat{y}$ directions are consolidated into

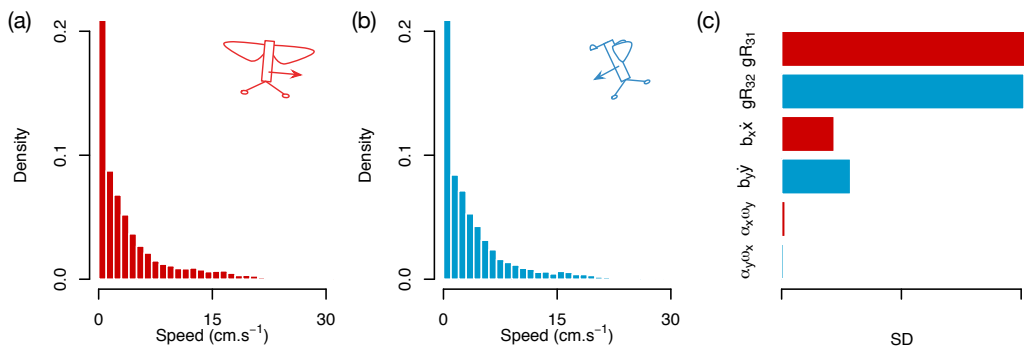

Figure 7.1: (a) Histogram showing the distribution of the flight speed data along the body $\hat{x}$-axis. (b) Histogram showing the distribution of the flight speed data along the body $\hat{y}$-axis. (c) Bar plot comparing the standard deviations of terms in the linear regression demonstrates the relative significance of each term.

one regression equation with the following components:

$$\begin{aligned} \Psi_i &= \begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\dot{x} + \omega_y\dot{z} - \omega_z\dot{y} & \text{for } \chi_i = \begin{bmatrix} -R_{31} & -\dot{x} & -\omega_y & 0 & 0 \end{bmatrix} \\ \frac{\mathrm{d}}{\mathrm{d}t}\dot{y} + \omega_z\dot{x} - \omega_x\dot{z} & \text{for } \chi_i = \begin{bmatrix} -R_{32} & 0 & 0 & -\dot{y} & \omega_x \end{bmatrix} \end{cases} \\ \varphi &= \begin{bmatrix} g & b_x & \alpha_x & b_y & \alpha_y \end{bmatrix}^T . \end{aligned} \tag{7.11}$$

It turns out that, in order to achieve reliable estimations of the parameters, particularly for the damping coefficients b_x and b_y, it is required that the velocity terms are relatively large in a significant portion of the flight data. Subsequently, in addittion to hovering and gradual maneuvers, we performed more experiments with the robot following circular trajectories at various flight speeds. In total, we obtain more than 126 seconds of data (15,120 flapping strokes) from 14 flights. The dataset used for the regression of the lateral dynamics is visualized in figure 7.1(a)-(b), which illustrates the proportion of observations recorded with various flight speeds along the body $\hat{x}$

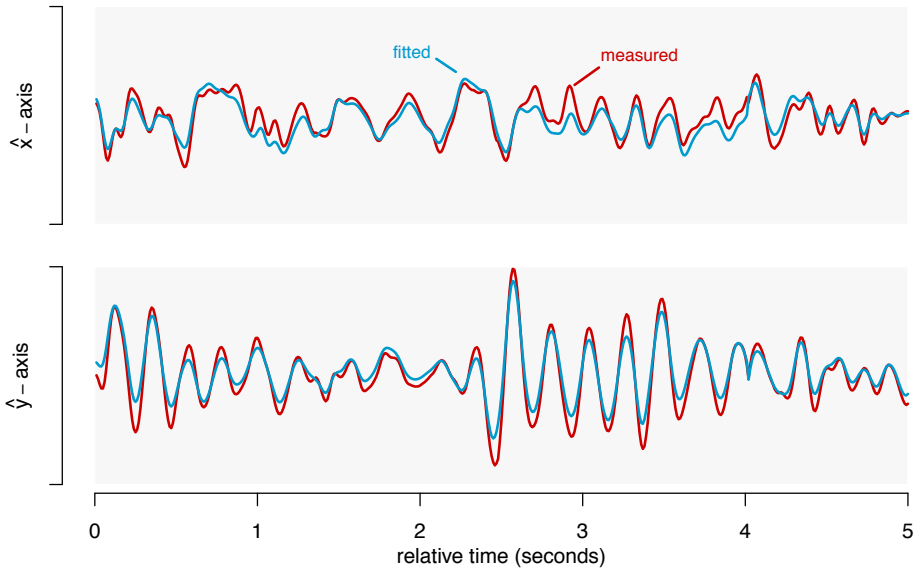

Figure 7.2: Comparison between a portion of the observation data Ψ and its corresponding fitted model $\chi\hat{\varphi}$ for the lateral dynamics according to equation (7.11). One linear regression is performed for both $\hat{x}$ and $\hat{y}$ axes, but they are plotted separately for illustrative purposes.

and $\hat{y}$ directions. The measured flight velocities are as large as 20 cm·s^{-1} at some instances, with a significant portion being larger than 5 cm·s^{-1}.

Estimates of parameters from the regression are listed in table 7.1. Selected examples of the observations and the model predictions are shown in figure 7.2. Table 7.1 reveals that the proposed linear model fits the observations well with the R^2 value of 0.79. We intentionally included the gravity term into the regression as a tool to evaluate the validity of the fit. In this case, the prediction of the gravitational constant g is found to be within 2% of the known value. The damping coefficients b_x and b_y are given as 2.1 s^{-1} and 3.1 s^{-1} respectively, consistent with the theoretical prediction (≈ 6 s^{-1}) for a robot with a similar (but not identical) geometry in [75]. Here it was found that the damping in the $\hat{y}$ direction is somewhat larger than in the $\hat{z}$ direction, which is intuitively anticipated given the robot geometry as defined in figure 2.5.

Figure 7.1(c) provides an illustration of the relative importance of each term in the regression by comparing the magnitude of the standard deviation of each fitted term. It suggests that the gravity terms dominates the lateral dynamics, followed by the lateral damping terms $b_x\dot{x}$ and $b_y\dot{y}$. The damping terms from the rotation, which are $\alpha_x\omega_y$ and $\alpha_y\omega_x$, were found to be insignificant, suggesting that the estimated values of α_x and α_y are likely to be inaccurate. Theoretically, this could be ameliorated by including more observations with large angular velocities, similar to the approach we have taken to boost the significance of $b_x\dot{x}$ and $b_y\dot{y}$ terms. In practice, due to the limited range of rotation, excitation of angular velocities swiftly leads to instability in controlled flight, preventing the robot to fly continuously with large angular velocities. In related work [72, 24], α_x and α_y were found to be insignificant and neglected.

7.3.3 Altitude Dynamics

Similar to the lateral dynamics, the altitude dynamics are extracted from the third row from equation (7.6). This altitude dynamics also involve the normalized thurst Γ generated by the robot. As a consequence, linear regression variables can be written as:

$$\begin{aligned} \Psi_i &= \frac{\mathrm{d}}{\mathrm{d}t}\dot{z} + \omega_x\dot{y} - \omega_y\dot{x} \\ \chi_i &= \begin{bmatrix} \frac{T}{s\gamma^{-1}+1} & -R_{33} & -\dot{z} \end{bmatrix} \\ \varphi &= \begin{bmatrix} \mu & g & b_z \end{bmatrix}^T, \end{aligned} \tag{7.12}$$

where s is a Laplace variable, and we have assumed that the thurst produced by the robot Γ is a low-passed signal of the commanded thrust T, scaled by an unknown scaling factor μ (which is nominally expected to be unity) according to $\dot{\Gamma} = \gamma\left(\mu T - \Gamma\right)$, similar to the previous assumption stated in equation (5.4). The presence of the scaling factor μ takes care of the discrepancy between the presumed thrust as modeled in Chapter (2) and the actual thrust produced by the robot that may also vary between robots.

Similar to the lateral dynamics, to perform the regression on the altitude dynamics, we design additonal trajectories that command the robot to follow sinusoidal trajectories along the vertical axis to increase the number of data points with non-zero altitude velocity. The dataset for the altitude regression contains five flights, totalling 54 seconds of flying time, which equates to 6,480 flapping periods. Figure 7.3(b) confirms that a large proportion of data has non-zero altitude speed. Multiple regressions were carried out at different values of γ^{-1} to determine the value that results in the best fit. The value of γ^{-1} represents the time scale of the actuation delay from the commanded thrust input to the thrust produced by the robot through the flapping-wing mechanism. To be precise, this number includes a partial delay in the experimental setup, the actuator response, and the aerodynamic response to the flapping wings. According to figure 7.3(a), the regression best fits the data when this delay is 10 ms. To put this number into perspective, it is comparable to one flapping period (8 ms), or the delay of the hardware loop in experimental setup of 8 ms as characterized in Appendix A.

Assuming γ^{-1} is 10 ms, the fitted parameters can be found in table 7.1. The

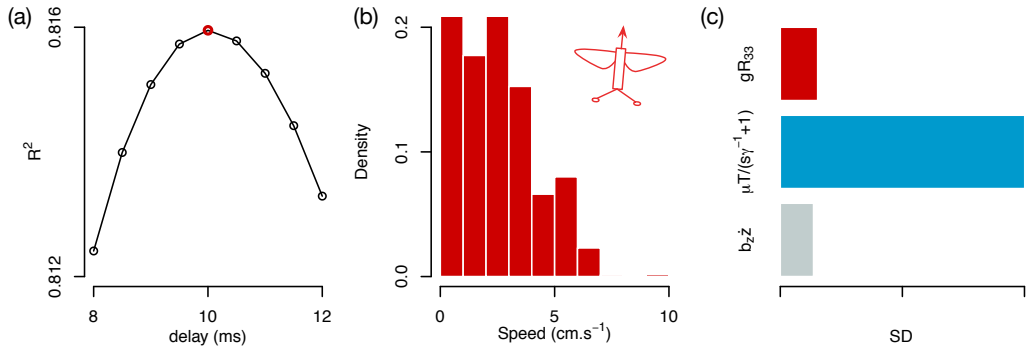

Figure 7.3: Regression of the altitude dynamics. (a) R^2 values obtained from multiple values of γ^{-1} (expressed as delay). The result shows that the model best fits the data when γ^{-1} is 10 ms. (b) The distribution of the speed along the $\hat{z}$ axis of the robot. (c) The plot showing the relative importance of terms in the regression of the altitude dynamics.

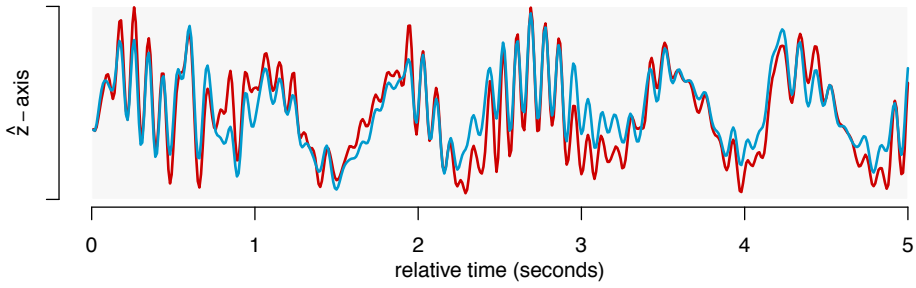

Figure 7.4: An example showing a portion of measurement data and fitted data in the regression of the altitude dynamics as given in equation (7.12).

goodness of fit, or R^2, is 0.82, indicating a reliable fit. The significance of terms are demonstrated in figure 7.3(c). The plot reveals that the thrust input, $\mu T/\left(s\gamma^{-1}+1\right)$, is the dominating term. The regression suggests that the gravitational constant is approximately 7.8 $\mathrm{m{\cdot}s^{-2}}$, 20% off the true value of 9.8 $\mathrm{m{\cdot}s^{-2}}$. This is relatively accurate given that the significance of the gravity term is considerably smaller than the dominant term as outlined in figure 7.3(c). Table 7.1 also provides a reasonable estimate of the damping coefficient b_z as 1.2 $\mathrm{m{\cdot}s^{-1}}$, somewhat smaller than the damping coefficients along the two lateral axes found earlier. An example of the observation and the fitted model of the altitude dynamic is demonstrated in figure 7.4.

7.3.4 Rotational Dynamics

The rotational dynamics given in equation (7.8) are slightly more sophisticated than the translational dynamics primarily owing to the coupling of the moment of inertia between axes. However, those terms could be lumped together and simplified into ν_i's and κ_i's as presented in equations (7.13) and (7.14). In this case, ν_i's, for example, account for the inverse of the moment of inertia along its corresponding axis multiplied by an unknown scaling factor that relates the assumed command torque to the actual produced torque. Moreover, we have introduced the factor $\left(s\gamma^{-1}+1\right)^{-1}$ to the torque commands to reflect the response time of the system in the same way

the system responds to the thurst command as found in Section 7.3.3.

$$\begin{aligned} \Psi_i &= \frac{\mathrm{d}}{\mathrm{d}t}\omega_x \\ \chi_i &= \begin{bmatrix} \tau_{c,x}\left(s\gamma^{-1}+1\right)^{-1} & \omega_y\omega_z & -\omega_x & \dot{y} \end{bmatrix} \\ \varphi &= \begin{bmatrix} \nu_x & \kappa_x & c_x & \beta_x \end{bmatrix}^T \end{aligned} \tag{7.13}$$

$$\begin{aligned} \Psi_i &= \frac{\mathrm{d}}{\mathrm{d}t}\omega_y \\ \chi_i &= \begin{bmatrix} \tau_{c,y}\left(s\gamma^{-1}+1\right)^{-1} & -\omega_x\omega_z & -\omega_y & -\dot{x} \end{bmatrix} \\ \varphi &= \begin{bmatrix} \nu_y & \kappa_y & c_y & \beta_y \end{bmatrix}^T \end{aligned} \tag{7.14}$$

Equations (7.13) and (7.14), which do not include the disturbance term d, serve as foundations for two regressions, each along one rotational axis. Parameters in φ_i's correspond to the lumped physical parameters expressed in equation (7.9). We consolidated nine trajectories, covering 72 seconds or $8,640$ flapping strokes, into one dataset. Assuming the response time of the system γ^{-1} is identical to the thurst dynamics, the regressions obtained the R^2 values of 0.22 and 0.21 along the $\hat{x}$ axis and $\hat{y}$ axis. We believe the poor fit is due to neglecting the disturbance term d that turns out to be significant in the case of the rotational dynamics. After careful inspection of relevant signals, it seems that the intrinsic dynamics of the unaccounted disturbance is relatively slow. As a consequence, it can be eliminated by a high-pass filter. We applied a high-pass filter with the cut off frequency of 2.0 Hz to all the terms in equations (7.13) and (7.14) and re-attempt the fitting process. The post-

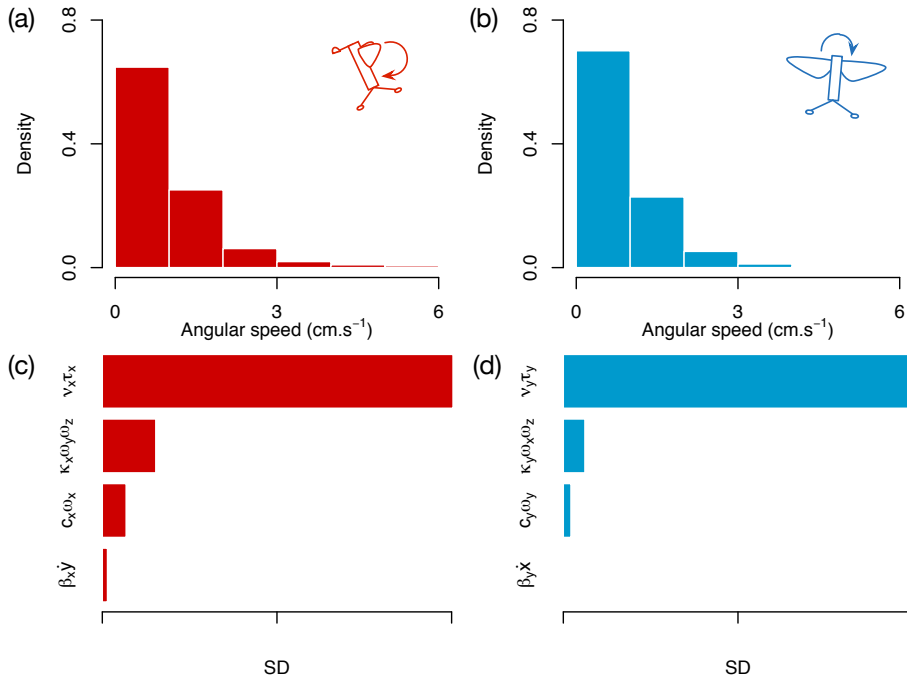

Figure 7.5: (a) Histogram illustrating the distribution of the angular velocity of the data along the pitch axis. (b) Histogram illustrating the distribution of the angular velocity of the data along the roll axis. (c) Bar plot demonstrating the relative significance of terms in the regression of the attitude data along the pitch axis. (d) Bar plot demonstrating the relative significance of terms in the regression of the attitude data along the roll axis.

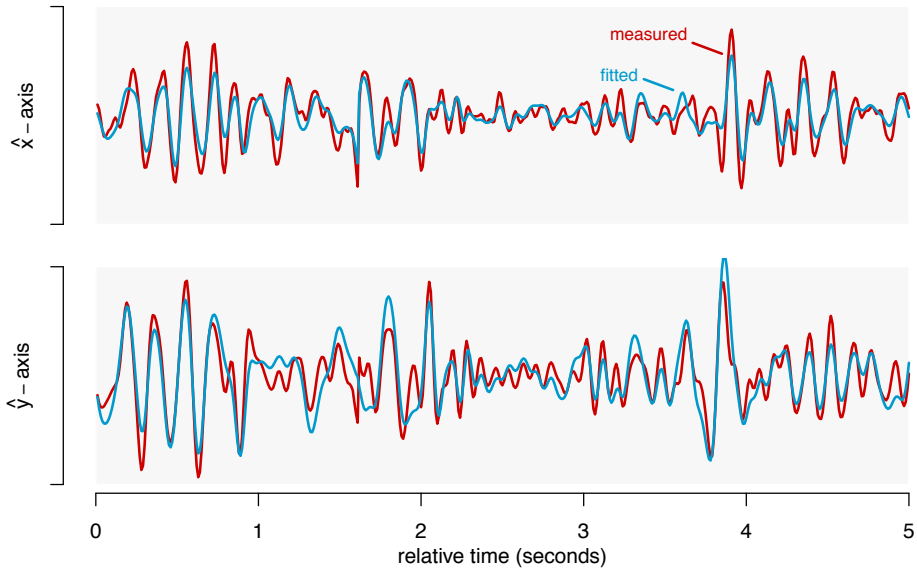

Figure 7.6: Examples of a portion of measurement data and fitted data in the regression of the attitude dynamics along the pitch ($\hat{x}$) axis and the roll $\hat{y}$ axis as given in equations (7.13) and (7.14).

filtered regressions yield the R^2 values of 0.83 and 0.73 along the $\hat{x}$ axis and $\hat{y}$ axis as tabulated in table 7.1, supporting the preposition that the disturbance only primarily the slow dynamics. We speculate that this disturbance is caused by the wire tether. More analysis on the effect of the tether can be found in Appendix B. Figure 7.5(c)-(d), unfortunately, shows that the rotational dynamics are primarily dominated by the torque commands, implying minimal contributions from the damping terms. This indicates that the estimated values of κ_i's, c_i's, and β_i's are unlikely to be accurate. Again, this is partially due to the limited range of angular velocities that could be generated and sustained during controlled flight as seen in figure 7.5(a)-(b).

The controller that was used to perform the flights in this section is presented in Chapter 5. The control law provided by equations (5.12) and (5.14) contains rotational damping terms, resembling the role of c_x and c_y in equations (7.13) and

(7.14). The controller gain was selected to ensure stability to satisfy tracking ability, causing it to dominate the rotational damping effect and undermine the influence of the aerodynamic damping in the analysis above. It is conceivable that the open-loop dynamics of the system may provide better estimates of the rotational damping coefficients. However, identification of open-loop dynamics is prevented by the instability of the system. Relevant work has mostly either obtained the rotational damping coefficient by a theoretical calculation [75], or from a dynamically-scaled robot that does not suffer from a rotation limit [23].

7.4 Static Experiments

To overcome the difficulty in estimating the rotational damping coefficient encountered in Section 7.3.4, in this section, we offer a theoretical approach to deduce the respective coefficients from the empirical flight data by analysing the sub-wingbeat dynamics of the robot.

To begin with, we revisit the rotational dynamics of the robot about the $\hat{x}$ axis

$$\sum \tau_x = J_x \omega_x + (J_z - J_y)\, \omega_y \omega_z.$$

At the time scale of one wingbeat (120 Hz), the torque acting on the wing, which may be described by equation (7.1), dominates the rotational dynamics, inducing the robot to rotate periodically about its $\hat{x}$ axis (the rotation about $\hat{y}$ and $\hat{z}$ axes are, on the other hand, negligible owing to the symmetry of the robot and the orientation of the flapping motion). Figure 7.7 plots the rotational rate of the robot ω_x, filtered to present only

Table 7.1: Estimates of parameters obtained from the linear regressions of flight dynamics. Estimates that are less certain (according to their relative significance) are displayed in grey.

Lateral dynamics		
R^2	0.79	
gravitational constant (g)	9.6	$\mathrm{m{\cdot}s^{-2}}$
b_x	2.1	$\mathrm{s^{-1}}$
b_y	3.1	$\mathrm{s^{-1}}$
α_x	8.5×10^{-3}	$\mathrm{m{\cdot}s^{-1}}$
α_y	-4.0×10^{-3}	$\mathrm{m{\cdot}s^{-1}}$
Altitude dynamics		
R^2	0.82	
γ^{-1}	10	ms
g	7.8	$\mathrm{m{\cdot}s^{-2}}$
μ	0.6	
b_z	1.2	$\mathrm{m{\cdot}s^{-1}}$
Rotational dynamics		
R^2 ($\hat{x}$-axis)	0.83	
ν_x	4.7×10^8	$\mathrm{kg^{-1}m^{-2}}$
κ_x	1.8	
c_x	-1.9	$\mathrm{s^{-1}}$
β_x	2.0	$\mathrm{m^{-1}s^{-1}}$
R^2 ($\hat{y}$-axis)	0.73	
ν_y	5.4×10^8	$\mathrm{kg^{-1}m^{-2}}$
κ_y	0.55	
c_y	-0.73	$\mathrm{s^{-1}}$
β_y	-0.13	$\mathrm{m^{-1}s^{-1}}$

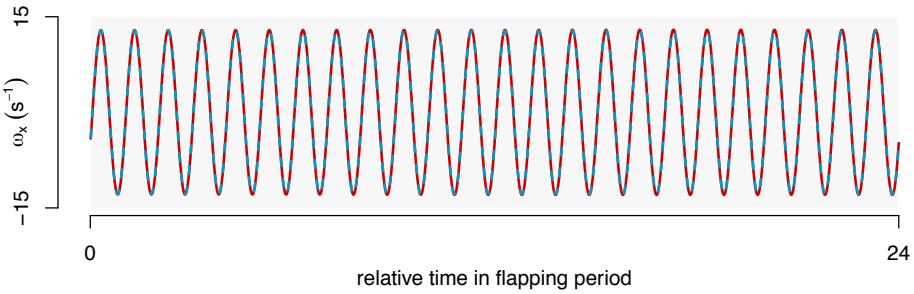

Figure 7.7: Comparision between the measured angular velocity of the robot along its pitch axis (red) and the fitted sinusoidal signal (blue).

the components near the flapping frequency. It is shown that the rotational rate could be accurately approximated as a sinusoidal function of the flapping frequency Ω as $\omega_x \approx \omega_{ox} \sin(\Omega t + \phi)$. This enables us to calculate the instantaneous torque produced by the flapping wings:

$$\begin{aligned} \tau &\approx J\frac{d}{dt}\omega_{ox}\sin(\Omega t + \phi) \\ &= J\Omega\omega_{ox}\cos(\Omega t + \phi) \end{aligned} \tag{7.15}$$

The instantaneous drag torque could be regarded as a product of the instantaneous drag force acting on the wing and the effective moment arm of length l according to

$$\tau(t) = D(t) \cdot l. \tag{7.16}$$

Focusing on the flapping frequency, figure 7.7 suggests that the robot experiences a maximum instantaneous angular velocity of 14.2 rad·s^{-1}, equating to a resultant drag torque from both wings of 12.9 μNm. By mounting the robot on a static force measurement setup and replicating the flapping-wing motion as illustrated in figure

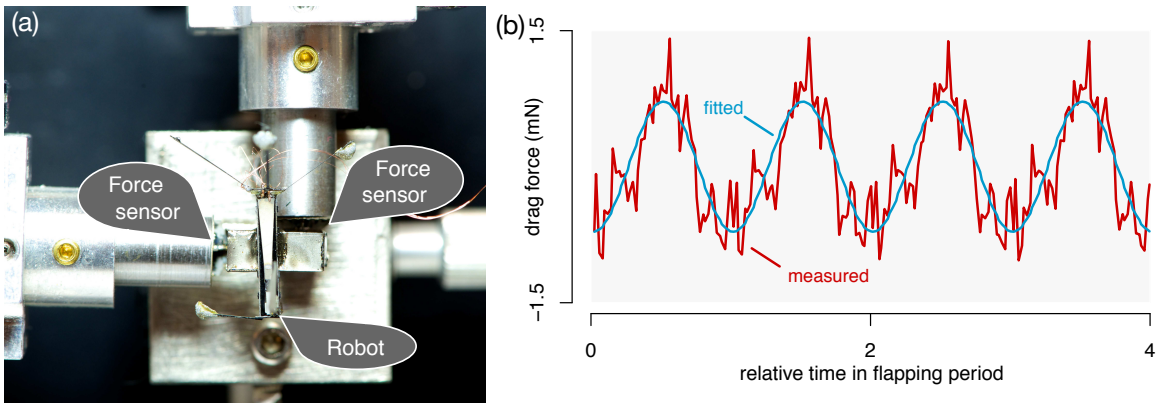

Figure 7.8: (a) The robot mounted upside-down on a compound beam structure connected to capacitive force sensors for lift and drag measurements. (b) Drag force measurement from the capacitive force sensor, fitted with a sinusoidal signal at the flapping frequency.

7.8(a), we could obtain the measurement of the instantaneous drag force from the high-precision capacitive force sensors. A portion of the measurement is shown in figure 7.8(b). The data is fitted with a sinusoidal signal at the wingbeat frequency of 120 Hz to eliminate the measurement noise and match the deduced dominant angular velocity information in figure 7.7. At the frequency of interest, the fit for drag from a single wing has an amplitude of 0.72 mN. The corresponding effective moment arm length l as found from equation (7.16) is, therefore, 9 mm. This number is likely to be larger then the true value, as the amplitude of the fitted signal is noticeably smaller than the unprocessed signal seen in figure 7.8(b). The predicted number for a similar robot given by [75] is 5 mm.

Based on equations (7.3) and (7.4), the rotational damping c and β could be approximated as $mb \cdot l^2/J$ and $mb \cdot l/J$ respectively. For the pitch axis, this yields $c_x = 15$ s^{-1} and $\beta_x = 1.6$ m^{-1}·ms^{-1}. These values are considerably larger than those corresponding to a similar-sized robot from the theoretical predictions provided by

[72], which were calculated to be 5 s^{-1} and 1 $m^{-1}{\cdot}ms^{-1}$. Nevertheless, it is worth noting that the dependence of c on l^2 means that any uncertainties in l would be amplified, resulting in a larger uncertainty in the deduced c, leading to the large discrepancy between the deduced value and the value reported in [72].

The rotational dynamics of the robot about its roll axis is vastly different from its pitch axis. The symmetry between the left wing and the right wing brings about no sub-wingbeat oscillation about the roll axis, preventing us from calculating the effective length of the moment arm using the same strategy. Under the assumption that the moment arm corresponding to the roll axis is similar to that of the pitch axis, we predict c_y and β_y to be 10 s^{-1} and 1.1 $m^{-1}{\cdot}ms^{-1}$.

7.5 Conclusion and Discussion

In previous chapters, some of the aerodynamics effects that arise in flight are neglected under the assumption that they could be regarded as disturbances by the flight controller. These unaccounted for effects could become increasingly important as the robot deviates from a perfect hovering condition. In this chapter, we thoroughly analyze the recorded flight trajectories from the experiments and construct physics-based models to gain further insights and establish more complete dynamic models of the robot. We find that, for instance, the robot encounters noticeable additional drag forces when it traverses laterally or vertically. On the other hand, the contribution from drag torques on the rotational dynamics is found to be less crucial as it is undermined by the flight controller. As a result, the rotational drag coefficients could not be reliably deduced from the flight data. Hence, we offer an alternative approach

to estimate the distance between the center of drag and the center of mass from the empirical data and predict the rotational drag coefficients based on theoretical approximations. It is anticipated that better dynamic models of the robot would eventually enhance the performace of flight controllers, reducing the tracking errors, particularly for more aggressive trajectories similar those presented in chapter 5 and 6.

7.5.1 Stability Analysis

In addition to the potential of enabling researchers to improve the control performance, information about damping and aerodynamic drag also provides better understanding about the dynamics of the robot and its stability. Stability of flapping-wing vehicles and flapping-wing insects has been widely discussed in literature [67, 75, 83, 82, 23]. Thus far, we have asserted that the flapping-wing robot employed in this thesis is inherently unstable, based on empirical evidence and comparison to insects of similar size.

To inspect the stability of the robot, we consider a linearized dynamic model of the robot in two dimensions, similar to the analyses presented in [75, 85]. The equations governing the longitudinal dynamics and the rotational dynamics are given in equation (7.17).

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\dot{x} &= g\theta - b\dot{x} \\ \frac{\mathrm{d}}{\mathrm{d}t}\dot{\theta} &= -c\dot{\theta} - \alpha\dot{x} + J^{-1}\tau, \end{aligned} \tag{7.17}$$

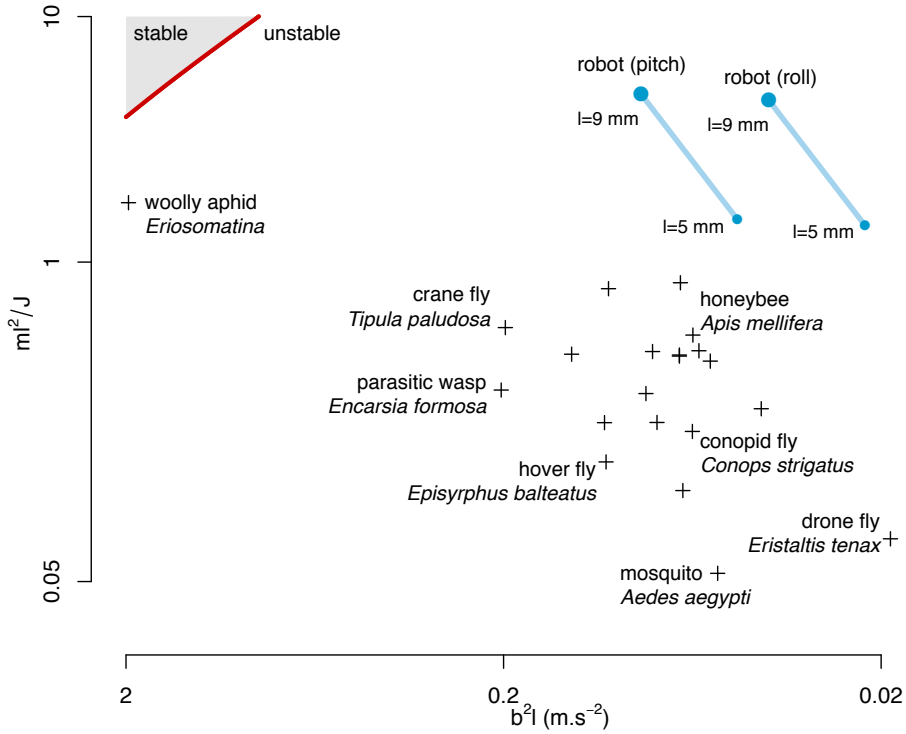

Figure 7.9: Relationship between the intrinsic stability and physical parameters of 21 insects and the robot used in this study. Physical parameters of insects are obtained from [75].

where we have carried over the notation from the previous sections. Here, the small angle approximation is applied, such that $\sin\theta \approx \theta$. To study the stability of the system under in the absence of the input torque, the longitudinal dynamic equation and the rotational dynamic equation can be merged to form a 3^{rd}-order ordinary differential equation:

$$\frac{\mathrm{d}^3}{\mathrm{d}t^3}\dot{x} + (b+c)\frac{\mathrm{d}^2}{\mathrm{d}t^2}\dot{x} + b\cdot c\frac{\mathrm{d}}{\mathrm{d}t}\dot{x} + \alpha\cdot g\dot{x} = 0 \tag{7.18}$$

Routh-Hurwitz stability criterion provides the sufficient conditions for the stability of

such a system. For the system of interest, where all damping coefficients are positive, the stability condition is satisfied when

$$\left(1 + m \cdot l^2/J\right) > \frac{1}{l \cdot b^2} g. \tag{7.19}$$

There are two quantities of interest in equation (7.19). First, $m \cdot l^2/J$, is a dimensionless quantity that is indicative of the geometry of the vehicle of the insect. It approximately measures the distance from the center of mass to the center of drag on the wing, normalized by the moment of inertia. The second quantity, which is $l \cdot b^2$, quantifies the strength of the drag compared to the gravitational constant. The product of these two quantities approximately determines the stability of the system. In the case of our robot, it is found that the robot is inherently unstable along both roll axis and pitch axis, over a wide range of possible l as illustrated in figure 7.9.

Additionally, figure 7.9 shows that 21 species of insects, including a number of bees and flies, are intrinsically unstable. The plot reveals that the dampling term $l \cdot b^2$ of the robot are comparable to that of insects, but the geometry and the distribution of mass (ml^2/J) are somewhat different. It seems that the elongated body of the robot, which brings about a relatively large l, renders the term ml^2/J larger than most insects. Nevertheless, the robot is still in the inherently unstable regime.

Chapter 8

Conclusion and Future Work

Using only tiny nervous systems, flying insects are able to perform superlative aerodynamic feats such as deftly avoiding a striking hand or landing on flowers buffeted by wind. This amazing maneuverability is unmatched by any man-made aircraft. These insects inspire scientists and engineers to understand and translate this ubiquitous form of locomotion into man-made machines. This resulted in flight-capable insect-scale prototypes first in [29, 58]. Yet, given their swift dynamics, inherent instability, and how little is known about flight at this scale, numerous steps had to be taken to bring such robots up in the air.

The significance of the results presented in this thesis are as follows: 1) For the insect-scale robot of interest, affine linear maps captured sufficient details between the input signals and the output torques for control purposes, despite the involvement of sophisticated nonlinear aerodynamics. 2) With an assumption that the attitude dynamics of the flapping-wing robot are considerably faster than the translational dynamics, we show that a simple nonlinear controller could stabilize the robot to

hover around a given setpoint, demonstrating the first controlled flight of an unstable flapping-wing robot that is truly insect-scale. 3) For MAVs at this scale, it is crucial to account for possible imperfections from the fabrication processes that variably and significantly affect the dynamics of the robots. This could be compensated by the use of an adaptive method. 4) In order to perform more dynamic trajectories as observed in natural fliers, the assumption that the attitude dynamics are much faster than the lateral dynamics has to be relaxed. We proposed an adaptive tracking controller that directly regulated the input signals based on position and orientation feedback, which could also be implemented on other types of flying vehicles. 5) Although, dynamics of extremely aggressive maneuvers such as perching on a vertical surface are highly nonlinear, the proposed feedback controller along with the iterative learning controller based on a simplified dynamic model was sufficient to enable to the robot to accomplish such a task. 6) Using the flight data from previous experiments, refined models of rotational and translational dynamics of the robot could be empirically deduced. The information is consistent with theoretical predictions and dynamically-scaled experiments in the literature.

Despite our pioneering work on the flight control of millimeter-scale flapping-wing robots, numerous issues have to be addressed before such robots can operate autonomously in outdoor environments as outlined below.

Onboard Sensor Integration

The presented control work relies immensely on the position and orientation feedback from the motion tracking system due to the lack of onboard sensors. At insect-

scale, the swift dynamics of the robot necessitates a lightweight and high-bandwidth sensing system. So far, bio-inspired solutions have been explored in ongoing work [21, 32, 41]. It was shown that a small optical-flow sensor was able to provide altitude feedback and an ocelli-inspired visual sensor was able to briefly stabilize the robot's upright orientation. However, more studies on alternative sensors such as gyroscopes and accelerometers are required to allow the robot to autonomously hover in place as achieved in other MAV systems [77, 48, 25, 49].

Miniaturization of Flight Electronics

In addition to sensors, power circuitry, flight electronics, and batteries must be incorporated into the robot to eliminate the tethering requirement in the current robot prototype. The design of flight electronics is likely dependent on the high-voltage and power requirements of the piezoelectric actuators, the choices of onboard sensors, and the available power source. Parallel work has been carried out and still ongoing on the topic of generating high-voltage signals from low power integrated circuits [46, 97].

Payload Capacity

The requirement to carry onboard sensors and flight electronic necessitates the robot to support additional payload. The current robot prototype was able to generate 50 mg of additional lift [58]. Unfortunately, this is unlikely to be sufficient for the extra components required to achieve autonomous flight. Preliminary calculations predict the total vehicle mass to be approximately twice as large as the current

prototype, with up to $30-40\%$ budgeted for a battery [47, 92]. The design and fabrication of a scaled-up prototype is unlikely to be straightforward owing to complicated underlying scaling laws [4, 89].

Disturbance Rejection for Outdoor Operations

In practice, as we transition to outdoor flight, it is crucial that the robot retain its stability in the presence of wind gust disturbances. It is known that wind is an important factor for flight [35]. Birds, for instance, are vulnerable to adverse weather [84]. The effect becomes increasingly more significant as the body size gets smaller. In insects, it has been observed that contributions from wind have to be correctly compensated for in their feedback system [33, 70]. The topic of wind disturbance rejection on other MAVs has been explored [87, 62]. At the insect scale, despite more severe effects, it is conceivable that similar studies could be carried out to enable the flapping-wing robot to operate in windy environments.

8.1 Concluding Remarks

Here we presented a series of models and controllers that successfully brought the millimeter-scale flapping-wing robot up in the air for the first time. As more was learned, the controllers gained more complexity, and accuracy. This eventually allowed the robot to execute highly dynamic and precise maneuvers like perching. However, there are still numerous challenges to be tackled by researchers before thousands of such robots will be able to roam around and functionally conduct operations such as environmental monitoring and crop pollination as we envision.

Bibliography

[1] M Achtelik, S Weiss, M Chli, and R Siegwart. Path planning for motion dependent state estimation on micro aerial vehicles. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2013.

[2] Markus W Achtelik, Simon Lynen, Margarita Chli, and Roland Siegwart. Inversion based direct position control and trajectory following for micro aerial vehicles. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 2933–2939. IEEE, 2013.

[3] Srinath Avadhanula, Robert J Wood, Erik Steltz, Joseph Yan, and Ronald S Fearing. Lift force improvements for the micromechanical flying insect. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1350–1356. IEEE, 2003.

[4] Adrian Bejan and James H Marden. Unifying constructal theory for scale effects in running, swimming and flying. *Journal of Experimental Biology*, 209(2):238–248, 2006.

[5] Frank M Bos, David Lentink, BW Van Oudheusden, and Hester Bijl. Influence of

wing kinematics on aerodynamic performance in hovering insect flight. *Journal of fluid mechanics*, 594(1):341–368, 2008.

[6] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. A survey of iterative learning control. *Control Systems, IEEE*, 26(3):96–114, 2006.

[7] J Caetano, J Verboom, C Visser, G Croon, B Remes, C Wagter, and M Mulder. Near-hover flapping wing MAV aerodynamic modelling-a linear model approach. *Int. J. Micro Air Vehicles*, 5(4), 2013.

[8] Bo Cheng, Xinyan Deng, and Tyson L Hedrick. The mechanics and control of pitching manoeuvres in a freely flying hawkmoth (manduca sexta). *The Journal of experimental biology*, 214(24):4092–4106, 2011.

[9] Pakpong Chirarattananon, Kevin Y Ma, and Robert J Wood. Adaptive control for takeoff, hovering, and landing of a robotic fly. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3808–3815. IEEE, 2013.

[10] Pakpong Chirarattananon and Robert J Wood. Identification of flight aerodynamics for flapping-wing microrobots. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1389–1396. IEEE, 2013.

[11] SA Combes, JD Crall, and S Mukherjee. Dynamics of animal movement in an ecological context: dragonfly wing damage reduces flight performance and predation success. *Biology letters*, 6(3):426–429, 2010.

[12] Rick Cory and Russ Tedrake. Experiments in fixed-wing uav perching. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2008.

[13] GCHE De Croon, KME De Clercq, R Ruijsink, B Remes, and C De Wagter. Design, aerodynamics, and vision-based control of the delfly. *International Journal of Micro Air Vehicles*, 1(2):71–97, 2009.

[14] GCHE de Croon, MA Groen, C De Wagter, B Remes, R Ruijsink, and BW van Oudheusden. Design, aerodynamics and autonomy of the delfly. *Bioinspiration & biomimetics*, 7(2):025003, 2012.

[15] Xinyan Deng, Luca Schenato, and S Shankar Sastry. Flapping flight for biomimetic robotic insects: Part ii-flight control design. *Robotics, IEEE Transactions on*, 22(4):789–803, 2006.

[16] Alexis Lussier Desbiens, Alan T Asbeck, and Mark R Cutkosky. Landing, perching and taking off from vertical surfaces. *The International Journal of Robotics Research*, 30(3):355–370, 2011.

[17] Michael H Dickinson. Haltere–mediated equilibrium reflexes of the fruit fly, drosophila melanogaster. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 354(1385):903–916, 1999.

[18] William B Dickson, Peter Polidoro, Melissa M Tanner, and Michael H Dickinson. A linear systems analysis of the yaw dynamics of a dynamically scaled insect model. *The Journal of Experimental Biology*, 213(17):3047–3061, 2010.

[19] William B Dickson, Andrew D Straw, Christian Poelma, and Michael H Dickinson. An integrative model of insect flight control. In *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*, 2006.

[20] Robert Dudley. *The biomechanics of insect flight: form, function, evolution*. Princeton University Press, 2002.

[21] P.-E.J. Duhamel, N.O. Perez-Arancibia, G.L. Barrows, and R.J. Wood. Biologically inspired optical-flow sensing for altitude control of flapping-wing microrobots. *Mechatronics, IEEE/ASME Transactions on*, 18(2):556–568, 2013.

[22] Reuven Dukas and Lauren Dukas. Coping with nonrepairable body damage: effects of wing damage on foraging performance in bees. *Animal Behaviour*, 81(3):635–638, 2011.

[23] Michael J Elzinga, Floris van Breugel, and Michael H Dickinson. Strategies for the stabilization of longitudinal forward flapping flight revealed using a dynamically-scaled robotic fly. *Bioinspiration & biomimetics*, 9(2):025001, 2014.

[24] Imraan Faruque and J Sean Humbert. Dipteran insect flight dynamics. part 1 longitudinal motion about hover. *Journal of Theoretical Biology*, 264(2):538–552, 2010.

[25] Festo. Festo SmartBird inspired by nature, April 2011.

[26] Benjamin M Finio, Kevin C Galloway, and Robert J Wood. An ultra-high precision, high bandwidth torque sensor for microrobotics applications. In *Intelligent*

Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pages 31–38. IEEE, 2011.

[27] Benjamin M Finio, Néstor Osvaldo Pérez-Arancibia, and Robert J Wood. System identification and linear time-invariant modeling of an insect-sized flapping-wing micro air vehicle. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1107–1114. IEEE, 2011.

[28] Benjamin M Finio and Robert J Wood. Distributed power and control actuation in the thoracic mechanics of a robotic insect. *Bioinspiration & Biomimetics*, 5(4):045006, 2010.

[29] Benjamin M Finio and Robert J Wood. Open-loop roll, pitch and yaw torques for a robotic bee. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 113–119. IEEE, 2012.

[30] Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience*, 5(7):1688–1703, 1985.

[31] Urban Forssell and Lennart Ljung. Closed-loop identification revisited. *Automatica*, 35(7):1215–1241, 1999.

[32] Sawyer B Fuller, Michael Karpelson, Andrea Censi, Kevin Y Ma, and Robert J Wood. Controlling free flight of a robotic fly using an onboard vision sensor inspired by insect ocelli. *Journal of The Royal Society Interface*, 11(97):20140281, 2014.

[33] Sawyer Buckminster Fuller, Andrew D Straw, Martin Y Peek, Richard M Murray, and Michael H Dickinson. Flying drosophila stabilize their vision-based velocity controller by sensing wind with their antennae. *Proceedings of the National Academy of Sciences*, 111(13):E1182–E1191, 2014.

[34] Jason Geder, Ravi Ramamurti, William Sandberg, and Anita Flynn. Modeling and Control Design for a Flapping-Wing Nano Air Vehicle. In *AIAA Guidance, Navigation, and Control Conference*, Reston, Virigina, June 2012. American Institute of Aeronautics and Astronautics.

[35] Frank B Gill. *Ornithology*. Macmillan, 1995.

[36] Karl G Götz and Christian Wehrhahn. Optomotor control of the force of flight in drosophila and musca. *Biological cybernetics*, 51(2):129–134, 1984.

[37] Jared Grauer, Evan Ulrich, James E Hubbard, Darryll Pines, and J Sean Humbert. Testing and system identification of an ornithopter in longitudinal flight. *Journal of Aircraft*, 48(2):660–667, 2011.

[38] Slawomir Grzonka, Giorgio Grisetti, and Wolfram Burgard. A fully autonomous indoor quadrotor. *Robotics, IEEE Transactions on*, 28(1):90–100, 2012.

[39] Ivar Gustavsson, Lennart Ljung, and Torsten Söderström. Identification of processes in closed loop-identifiability and accuracy aspects. *Automatica*, 13(1):59–75, 1977.

[40] Elliot W Hawkes, David L Christensen, Eric V Eason, Matthew A Estrada, Matthew Heverly, Evan Hilgemann, Hao Jiang, Morgan T Pope, Aaron Parness,

and Mark R Cutkosky. Dynamic surface grasping with directional adhesion. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5487–5493. IEEE, 2013.

[41] E Farrell Helbling, Sawyer B Fuller, and Robert J Wood. Pitch and yaw control of a robotic insect using an onboard magnetometer. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014.

[42] Warren Hoburg and Russ Tedrake. System identification of post stall aerodynamics for uav perching. In *Proceedings of the AIAA Infotech@ Aerospace Conference*, 2009.

[43] Gabriel M Hoffmann, Haomiao Huang, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference*, volume 2, 2007.

[44] Mu Huang, Bin Xian, Chen Diao, Kaiyan Yang, and Yu Feng. Adaptive tracking control of underactuated quadrotor unmanned aerial vehicles via backstepping. In *American Control Conference (ACC), 2010*, pages 2076–2081. IEEE, 2010.

[45] John David Jackson. Classical electrodynamics. *Classical Electrodynamics, 3rd Edition, by John David Jackson, pp. 832. ISBN 0-471-30932-X. Wiley-VCH, July 1998.*, 1, 1998.

[46] Michael Karpelson, Gu-Yeon Wei, and Robert J Wood. Driving high voltage

piezoelectric actuators in microrobotic applications. *Sensors and Actuators A: Physical*, 176:78–89, 2012.

[47] Michael Karpelson, John Peter Whitney, Gu-Yeon Wei, and Robert J Wood. Energetics of flapping-wing robotic insects: towards autonomous hovering flight. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1630–1637. IEEE, 2010.

[48] Matthew Keennon, Karl Klingebiel, Henry Won, and Alexander Andriukov. Development of the nano hummingbird: A tailless flapping wing micro air vehicle. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, pages 1–24, 2012.

[49] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[50] Vladislav Klein and Eugene A Morelli. *Aircraft system identification: theory and practice*. American Institute of Aeronautics and Astronautics Reston, VA, USA, 2006.

[51] Mirko Kovač, Jürg Germann, Christoph Hürzeler, Roland Y Siegwart, and Dario Floreano. A perching mechanism for micro aerial vehicles. *Journal of Micro-Nano Mechatronics*, 5(3-4):77–91, 2009.

[52] Daewon Lee, H Jin Kim, and Shankar Sastry. Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter. *International Journal of control, Automation and systems*, 7(3):419–428, 2009.

[53] Taeyoung Lee, M Leoky, and N Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE(3). In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 5420–5425. IEEE, 2010.

[54] Fernando Lizarralde and John T Wen. Attitude control without angular velocity measurement: A passivity approach. *Automatic Control, IEEE Transactions on*, 41(3):468–472, 1996.

[55] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

[56] Sergei Lupashin, Angela Schöllig, Michael Sherback, and Raffaello D'Andrea. A simple learning strategy for high-speed quadrocopter multi-flips. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1642–1648. IEEE, 2010.

[57] Kevin Y Ma, Pakpong Chirarattananon, Sawyer B Fuller, and Robert J Wood. Controlled flight of a biologically inspired, insect-scale robot. *Science*, 340(6132):603–607, 2013.

[58] Kevin Y Ma, Samuel M Felton, and Robert J Wood. Design, fabrication, and modeling of the split actuator microrobotic bee. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1133–1140. IEEE, 2012.

[59] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and

control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[60] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 31(5):664–674, 2012.

[61] Yiğit Mengüç, Michael Röhrig, Uyiosa Abusomwan, Hendrik Hölscher, and Metin Sitti. Staying sticky: contact self-cleaning of gecko-inspired adhesives. *Journal of The Royal Society Interface*, 11(94):20131205, 2014.

[62] Abdellah Mokhtari and Abdelaziz Benallegue. Dynamic feedback controller of euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 3, pages 2359–2366. IEEE, 2004.

[63] Mark Muller, Sergei Lupashin, and Raffaello D'Andrea. Quadrocopter ball juggling. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 5113–5120. IEEE, 2011.

[64] Andrew Y Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer, 2006.

[65] Michael W Oppenheimer, David B Doman, and David O Sigthorsson. Dynamics and control of a biomimetic vehicle using biased wingbeat forcing functions:

Part i-aerodynamic model. In *Proceedings of the 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Exposition, Orlando, Florida, USA*, 2010.

[66] Michael W Oppenheimer, David B Doman, and David O Sigthorsson. Dynamics and control of a biomimetic vehicle using biased wingbeat forcing functions. *Journal of guidance, control, and dynamics*, 34(1):204–217, 2011.

[67] Christopher T Orlowski and Anouck R Girard. Dynamics, stability, and control analyses of flapping wing micro-air vehicles. *Progress in Aerospace Sciences*, 51:18–30, 2012.

[68] P Paoletti and L Mahadevan. Intermittent locomotion as an optimal control strategy. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, 470(2164):20130535, 2014.

[69] Néstor O Pérez-Arancibia, Kevin Y Ma, Kevin C Galloway, Jack D Greenberg, and Robert J Wood. First controlled vertical flight of a biologically inspired microrobot. *Bioinspiration & Biomimetics*, 6(3):036009, 2011.

[70] Michael B Reiser, J Sean Humbert, Mary J Dunlop, Domitilla Del Vecchio, Richard M Murray, and Michael H Dickinson. Vision as a compensatory mechanism for disturbance rejection in upwind flight. In *American Control Conference, 2004. Proceedings of the 2004*, volume 1, pages 311–316. IEEE, 2004.

[71] Charles Richter and Hod Lipson. Untethered hovering flapping flight of a 3d-printed mechanical insect. *Artificial life*, 17(2):73–86, 2011.

[72] Leif Ristroph, Attila J Bergou, Gordon J Berman, John Guckenheimer, Z Jane Wang, and Itai Cohen. Dynamics, control, and stabilization of turning flight in fruit flies. In *Natural locomotion in fluids and on surfaces*, pages 83–99. Springer, 2012.

[73] Leif Ristroph, Attila J Bergou, John Guckenheimer, Z Jane Wang, and Itai Cohen. Paddling mode of forward flight in insects. *Physical review letters*, 106(17):178103, 2011.

[74] Leif Ristroph and Stephen Childress. Stable hovering of a jellyfish-like flying machine. *Journal of The Royal Society Interface*, 11(92):20130992, 2014.

[75] Leif Ristroph, Gunnar Ristroph, Svetlana Morozova, Attila J Bergou, Song Chang, John Guckenheimer, Z Jane Wang, and Itai Cohen. Active and passive stabilization of body pitch in insect flight. *Journal of The Royal Society Interface*, 10(85):20130237, 2013.

[76] Angela Schöllig and Raffaello D'Andrea. Optimization-based iterative learning control for trajectory tracking. In *Proceedings of the European control conference (ECC)*, pages 1505–1510, 2009.

[77] Shaojie Shen, Nathan Michael, and Vijay Kumar. Autonomous multi-floor indoor navigation with a computationally constrained mav. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 20–25. IEEE, 2011.

[78] Wei Shyy, Hikaru Aono, Satish Kumar Chimakurthi, P Trizila, C-K Kang, Car-

los ES Cesnik, and Hao Liu. Recent progress in flapping wing aerodynamics and aeroelasticity. *Progress in Aerospace Sciences*, 46(7):284–327, 2010.

[79] Wei Shyy, Yongsheng Lian, Jian Tang, Dragos Viieru, and Hao Liu. *Aerodynamics of low Reynolds number flyers*. Cambridge University Press New York, 2008.

[80] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*, volume 199. Prentice hall New Jersey, 1991.

[81] Pratheev S Sreetharan, John P Whitney, Mark D Strauss, and Robert J Wood. Monolithic fabrication of millimeter-scale machines. *Journal of Micromechanics and Microengineering*, 22(5):055027, 2012.

[82] Mao Sun, Jikang Wang, and Yan Xiong. Dynamic flight stability of hovering insects. *Acta Mechanica Sinica*, 23(3):231–246, 2007.

[83] Graham K Taylor. Mechanics and aerodynamics of insect flight control. *Biological Reviews*, 76(4):449–471, 2001.

[84] Hendrik Tennekes. *The simple science of flight: from insects to jumbo jets*. MIT press, 2009.

[85] Zhi Ern Teoh, Sawyer B Fuller, Pakpong Chirarattananon, NO Prez-Arancibia, Jack D Greenberg, and Robert J Wood. A hovering flapping-wing microrobot with altitude control and passive upright stability. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3209–3216. IEEE, 2012.

[86] Gábor Vásárhelyi, Csaba Virágh, Gergő Somorjai, Norbert Tarcai, Tamás Szörényi, Tamás Nepusz, and Tamás Vicsek. Outdoor flocking and formation flight with autonomous aerial robots. *arXiv preprint arXiv:1402.3588*, 2014.

[87] Steven L Waslander and Carlos Wang. Wind disturbance estimation and rejection for quadrotor position control. In *AIAA Infotech@ Aerospace Conference and AIAA Unmanned... Unlimited Conference, Seattle, WA*, 2009.

[88] JP Whitney and RJ Wood. Aeromechanics of passive rotation in flapping flight. *Journal of Fluid Mechanics*, 660:197–220, 2010.

[89] JP Whitney and RJ Wood. Conceptual design of flapping-wing micro air vehicles. *Bioinspiration & biomimetics*, 7(3):036001, 2012.

[90] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[91] RJ Wood, E Steltz, and RS Fearing. Optimal energy density piezoelectric bending actuators. *Sensors and Actuators A: Physical*, 119(2):476–488, 2005.

[92] Robert J Wood. The first takeoff of a biologically inspired at-scale robotic insect. *Robotics, IEEE Transactions on*, 24(2):341–347, 2008.

[93] Robert J Wood, Ben Finio, Michael Karpelson, Kevin Ma, Néstor Osvaldo Pérez-Arancibia, Pratheev S Sreetharan, Hiroto Tanaka, and John Peter Whitney. Progress on 'pico' air vehicles. *The International Journal of Robotics Research*, 31(11):1292–1302, 2012.

[94] Rong Xu and Umit Ozguner. Sliding mode control of a quadrotor helicopter. In *Decision and Control, 2006 45th IEEE Conference on*, pages 4957–4962. IEEE, 2006.

[95] Joseph Yan, Robert J Wood, Srinath Avadhanula, Metin Sitti, and Ronald S Fearing. Towards flapping wing control for a micromechanical flying insect. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 3901–3908. IEEE, 2001.

[96] JM Zanker. On the mechanism of speed and altitude control in drosophila melanogaster. *Physiological entomology*, 13(3):351–361, 1988.

[97] Xuan Zhang, David Brooks, and Gu-Yeon Wei. A 20μw 10mhz relaxation oscillator with adaptive bias and fast self-calibration in 40nm cmos for micro-aerial robotics application. In *Solid-State Circuits Conference (A-SSCC), 2013 IEEE Asian*, pages 433–436. IEEE, 2013.

Appendix A

Latency of the Experimental Setup

A.1 Introduction

To quantify the latency of the experimental system, including the processing time of the motion capture system and the communication time between computing units, a simple high-bandwidth actuated system—an unloaded bimorph piezoelectric actuator with retroreflective markers attached for tracking the tip deflection—was driven using low frequency sinusoidal signals. The motion of the actuated link was tracked by the motion capture system. The information was then fed back into the controller. The time difference between the command signal and the resultant motion, after subtracted away the delay of the mechanical actuated link, represents the latency of the experimental setup.

A.2 Experimental Methods

The actuated system was driven by sinusoidal signals with frequencies ranging from 0.5 Hz to 2.0 Hz, three trials were executed at each frequencty. This was carried out using the xPC target environment at the rate of 10 kHz. The motion capture system then tracked and recorded the position of three retroreflective markers and constructed the trajectory of the tip of the actuator at the rate of 500 Hz. The data is then smoothed using a cubic spline from 500 Hz to 2000 Hz.

The motion of the actuated system was assumed to follow closely a sinusoidal trajectory. To identify the time delay between the driving signal and the recorded trajectory, a gradient descent algorithm was used to fit a sinusoidal curve to the trajectory in a least-squares fashion. The time delay between the driving signal and the trajectory was then found by comparing two sinusoidal signals.

This calculated total time delay T constitutes of the delay caused by the mechanical system known as the phase shift ϕ and the external delay caused by the experimental setup Δ. At low frequencies, the phase shift is expected to be small. We assume that in this small range of frequencies we performed the experiments this phase shift is constant, regardless of the frequency. Therefore, the effect of the delay from the phase shift would be proportional to the period of the driving signal as captured by equation A.1:

$$T = \phi\frac{1}{f} + \Delta. \tag{A.1}$$

The external delay, Δ, we seek to determine, on the other hand, should remain constant independent of the driving frequency.

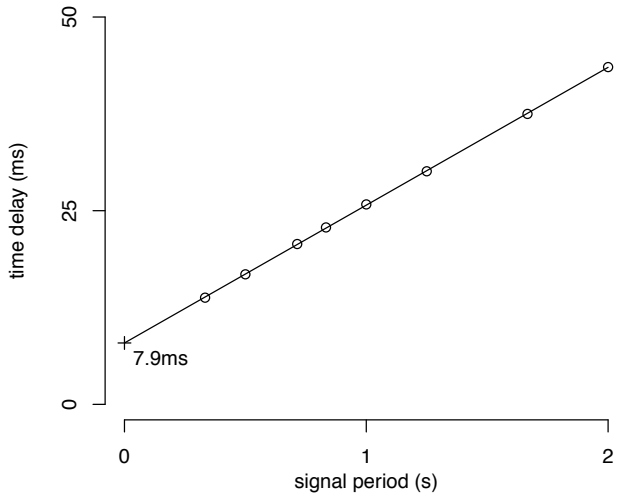

Figure A.1: Plot of the total time delay of a simple high-bandwidth actuated system at various driving frequencies. Each point is an average of three trials. The result suggests that the latency of the experimental setup is approximately 8 ms.

A.3 Results

Figure A.1 presents a relationship between the periods of the driving signals and the measured time delay obtained from the least-squares fit. The plot reveals a strong linear relationship. The linearity verifies the assumption that over this range of frequencies, the phase shift is approximately constant (and therefore, the resultant time delay is proportional to the signal period.). From the plot, the intersection of the best fit line to the $\hat{y}$ axis respresents the constant time delay as a result of the latency of the experimental setup. This is found to be 8 ms as shown in figure A.1.

For the robotic fly, the actuator-transmission-wing system could be approximated as a second order system [27]. Near its resonant frequency, it is expected that the flapping motion would be no more than 180° out of phase from the driving signals. This approximately equates to 4 ms delay for the flapping frequency of 120 Hz. In total, the latency of the experimental setup could be as long as 12 ms.

It is worth noting that the value of the latency of the setup depends on several factors, including the number of markers being tracked and the complexity of the executed algorithms. In this circumstance, only three retroreflective makers were tracked, whereas four were usually used in flight control experiments. Moreover, no controller was executed in this test. This suggests that the actual latency of the experimental setup could be slightly larger than the value obtained here.

Appendix B

Effects of the Wire Tether

B.1 Introduction

In the current prototype of the flapping wing robot that was first introduced in chapter 2, its small scale and limited payload capacity prevent the robot from carrying flight avionics, sensors, and batteries. The robot is therefore tethered for power, sensing, and computation through a bundle of four 51-gauge wires. The thin profile of 51-gauge wires and their light weight allows the robot to have stable unconstrained flight under controlled conditions. The effects of the tether on the translational and rotational dynamics of the robot have been presumed to be negligible. However, this has never been systematically quantified or well understood owing to the unpredictable nature of the tether.

To obtain better understanding and to put bounds on the effects of the wire tether on the translational and rotational dynamics of the robot, in this appendix we propose two theoretical models to cover two possible extreme scenarios. The first

model regards the tether is a rigid rod that rests between the robot and the floor. The second model assumes the tether to be completely flexible, similar to a string that cannot support any compressive force. In reality, it is anticipated that the behavior of the partially stiff tether falls somewhere between the two proposed models. As a result, we will assume that the actual contribution from the tether would also be between the bounds given by both models.

In the last section of this appendix, we analyze the flight data and inspect the parts of the commanded torques that are believed to be counteracting the disturbances arising from the tether wire. Lasly, this data is compared to the model predictions as a validating process.

B.2 Theoretical Models

B.2.1 Rigid rod model

In this model, the tether wire of length L is treated as a rigid rod of weight W that is leaning between the robot and the floor as depicted in figure B.1(a). Five forces acting on the rod are labeled. At point A, the tether is lifted (N_1) and pushed (F) by the robot. For a hovering robot, the tether is assumed to be in equilibrium, such that $W = N_1 + N_2$. We can also obtain the moment about the point B, where the tether is resting on the floor, as

$$\frac{1}{2}WL\cos\theta \;\; = \;\; N_1 L\cos\theta + FL\sin\theta,$$

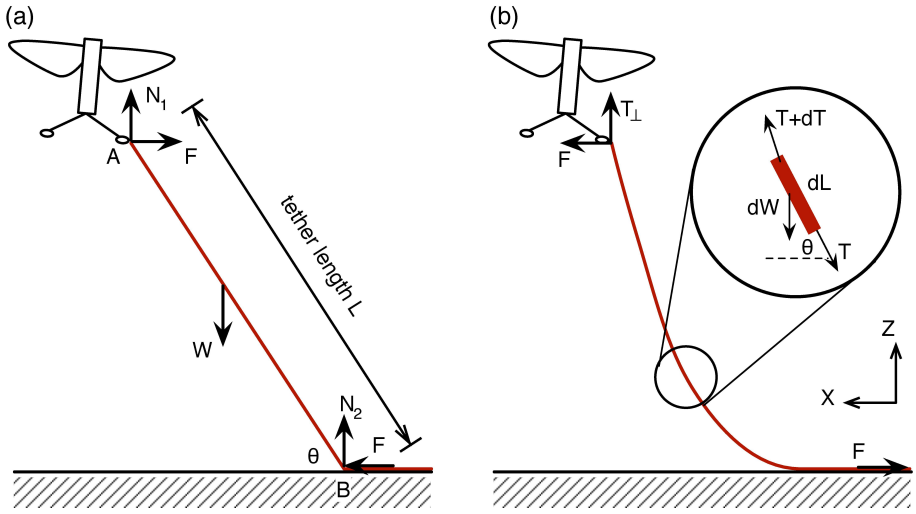

Figure B.1: Schematic diagrams illustraing the free-body diagram of the wire tether in two simulated scenarios. (a) The rigid rod model. (b) The flexible string model.

which can be re-arranged in to

$$\frac{1}{2}W = N_1 + F\tan\theta. \tag{B.1}$$

Without the knowledge of the coefficient of friction between the tether and the floor, it is impossible to find an explicit expresion of N_1 or F. Nevertheless, equation (B.1) allows us to bound them from above:

$$\begin{aligned} \|N_1\| &\leq \frac{1}{2}W \\ \|F\| &\leq \frac{1}{2\tan\theta}W. \end{aligned}$$

Empirically, it is reasonable to assume that $\theta > 45°$, rendering $\|F\| \leq W/2$. For a tether of length 15 cm, it was found to weigh ≈ 3 mg, or ≈ 30 μN. Therefore, the upper-bounds of N_1 and F approximately equate to 2% of the robot's weight. Their

contribution to the translational dynamics of the robot is negligible.

To evaluate the impact of the wire tether to the rotational dynamics of the robot, we assume that the tether is attached to the robot one centimeter from its center of mass. When combined, N_1 and F may exert up to 0.3 μNm torque on the robot. While this is somewhat smaller than the maximum torque the robot can generate (on the order of 1 μNm), it could potentially have significant effect on the rotational dynamics. Since, the dynamics of the tether is largely determined by the position of the robot, or the translational dynamics of the robot, it is considerably slower than the rotational dynamics. The contribution from the tether is, therefore, aproximately constant in the timescale of the rotational dynamics. A constant disturbance torque from the tether would result in a destabilizing effect on the robot along the roll or pitch direction, presuming the rigid rod model is a good representation of the reality.

B.2.2 Flexible string model

The opposite extreme of the rigid rod model is a flexible string model, in which we regard the wire tether as a string that can only be in tension, not compression. We suspect that in practice the tether behaves more similarly to this model than to the previous model based on some experimental observations of the robot in unconstrained flight. Similar to the preious model, it is assumed that the tether is approximately in equilibrium. A diagram elaborating the flexible string model is shown in figure B.1(b). We begin the analysis by considering the tether at the point where it is resting on the floor at $X = 0$ and $Z = 0$. At this point, the tension in the tether is only in the horizontal direction.

The flexibility assumption implies that the shape of the tether at any point conforms to the tension at that point. The tension along the horizontal direction is constant throughout since there is no external force acting on the tether in that direction. In contrast, the tension in the vertical direction varies as the weight of the tether distributes uniformly along itself. Consider an infinitesimal section of the tether of length dL, an increase in the vertical component of the tension, $dT_\perp$, in the section is given by $\rho g dL$, where ρ is the linear density of the tether. It follows that we can relate the small change in the tension to the shape (or angle) of the tether at a point as

$$d\tan\theta = \frac{dT_\perp}{F} = \frac{\rho g dL}{F}. \tag{B.2}$$

Integration of equation (B.2) enables us to find the vertical component of the tether and angle of the tether as a function of length from the floor.

$$\begin{aligned} T_\perp &= W \\ \tan\theta &= \frac{\rho g L}{F} = \frac{W}{F}, \end{aligned}$$

where we have substituted $\rho g L$ by the weight of the tether W. What remains unknown is the horizontal component of the tension, F. This can be found by analyzing the geometry of the tether. For a infinitesimal section of the tether positioned in the

$X - Z$ plane at an angle θ, we can write

$$\begin{aligned} dL &= \sqrt{dx^2 + dz^2} \\ &= \sqrt{1 + \cot^2\theta}dz = \sqrt{1 + \left(\frac{F}{\rho g L}\right)^2}dz. \end{aligned} \tag{B.3}$$

Integrating equation (B.3) from the floor $z = 0$, $L = 0$ to the robot $z = h$, $L = L$, we obtain equation (B.4).

$$h = \sqrt{L^2 + \left(\frac{F}{\rho g}\right)^2} - \frac{F}{\rho g} \tag{B.4}$$

This describes the relationship between the horizontal component of the tension F as a function of the height of the robot given the hanging length of the tether L. The intepretion is that the horizontal tension depends on the geometry of the tether, or the vertical position of the robot. The information of h or L could be extracted from flight videos. F could be calculated explicitly by solving equation (B.4).

$$F = \rho g \frac{(L^2 - h^2)}{2h}.$$

For $L = 15$ cm and $h = 12$ cm, this yields $F = 7$ μN. This is a few times smaller than the weight of the tether (30 μN) that produces the vertical component of the tension.

Again, the tension of the tether is significantly smaller than the weight of the robot itself, making it unlikely to provide noticeable impact on the translation dynamics of the robot. Yet, it may have a considerable influence on the rotational dynamics of the robot similar to the rigid rod model case. In this circumstance, however, the

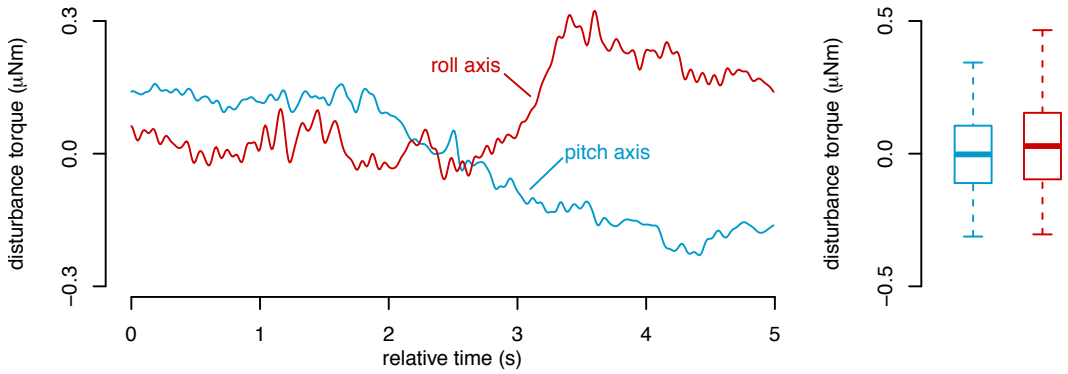

Figure B.2: Disturbance torque deduced from flight data. (left) A five-second portion of the presumed disturbance torques extracted from the flight data. (right) The boxplots showing the means, standard devations, and ranges of the disturbance torques from more than 72 seconds (8,640 wing beats) of flight.

direction of the horizontal component of the tension lies such that its effect would counteract its vertical component, making the total torque likely to be smaller than 0.3 μNm.

B.3 Experimental Observations

Previously in chapter 7 we initially failed to correlate the torque commands to the rotational dymamics of the robot. It was found to be crucial to filter out low-frequency components of the signal for the dynamic model to explain the flight data well. This finding could be explained if there existed unknown torque disturbance terms whose dynamics are mostly slower than 2.0 Hz. During flight, these disturbances are perceived and corrected by the controller. We believe the disturbances are primarily caused by the wire tether, as its effects are likely to be principally unpredictable but dependent on the position of the robot, and therefore, intrinsically slow.

To support the claim that the tether is likely to be the main cause of disturbances in the rotational dynamics, we analyze the low-frequency components of the commanded torques, which are believed to be in response to the disturbances. Over 72 seconds of flight data, these torques are found to average around zero, with the standard deviations of 0.10 μNm and 0.13 μNm along the pitch and roll axes as shown on the right hand side of figure B.2. Their extreme values are below 0.50 μNm. The magnitudes of the unknown disturbances are consistent with the predictions from both models (which turn out to be very similar) in the preceding section. These effects are considerably large for the robot whose the moment of inertial are on the order of 10^{-9} kg·m^2 and potentially can destabilize the robot. Fortunately, they were efficiently suppressed by the high gain and quick response of the controller.